

PRELIMINARY DRAFT COPY

CMMS Representations

PRELIMINARY DRAFT COPY

Table of Contents

1 Purpose	8
2 Executive Summary	8
3 Overview	9
3.1 What is the business of CMMS?	9
3.2 How to do CMMS	9
3.3 Objectives of CMMS	11
3.3.1 Correctness of Conceptual Models	12
3.3.2 Documentation of Conceptual Models	12
3.3.3 Utility of Conceptual Models	13
3.3.4 Consistency of Conceptual Models	13
3.3.5 Composability of Conceptual Models	13
3.3.6 Reusability of Conceptual Models	14
3.3.7 Produceability of Conceptual Models	14
4 CMMS Description	15
4.1 Introduction	15
4.2 CMMS Users	15
4.3 CMMS Knowledge Elements	16
4.4 A Scenario for CMMS Production	16

PRELIMINARY DRAFT COPY

4.5 A Process for CMMS	18
4.6 Three Model Views	20
4.7 Conceptual Model Framework	21
5 CMMS Knowledge Elements and Discussion	22
5.1 Use Case	23
5.2 Action	25
5.3 Task	27
5.3.1 Nominal Tasks	31
5.3.2 Substantive Tasks	31
5.4 Sequence	32
5.5 Condition	36
5.6 Use Case Instance, Condition Values, and MOP Standards	37
5.7 Entity	39
5.7.1 Rank	45
5.7.2 Classes	45
5.7.3 Objects	45
5.8 Association	46
5.9 Interaction	50
5.10 Verb	56

PRELIMINARY DRAFT COPY

5.11	Inheritance	56
5.12	Aggregation	56
5.13	Configuration Unit	56
5.14	Data Dictionary	59
5.15	Metadata	60
5.16	Enumeration	61
5.17	Examination	62
5.18	Reference	64
5.19	Conceptual Object Model	65
5.20	Attributes	66
5.21	Fidelity	67
5.22	State	68
	Appendix A. Glossary	69
	A-1. CMMS Glossary	69
	A-2. Document Usage Glossary	74
	A-3. Acronyms	74
	Appendix B. References	76
	Appendix C. Environment	76
	C-1. Physical Environment	77

PRELIMINARY DRAFT COPY

C-1.1. Land	77
C-1.2. Sea	77
C-1.3. Air	78
C-1.3.1 Weather	78
C-1.4. Space	79
C-2. Military Environment	79
C-2.1. Mission	79
C-2.2. Forces	79
C-2.3. Command, Control, and Communications	80
C-2.4. Intelligence	80
C-2.5. Deployment, Movement, and Maneuver	80
C-2.6. Firepower	80
C-2.7. Protection	80
C-2.8. Sustainment	81
C-2.9. Threat	81
C-3. Civil Environment	81
C-3.1. Political Policies	81
C-3.2. Culture	81
C-3.3. Economy	82
Appendix D. Entity Examples	82

PRELIMINARY DRAFT COPY

Appendix E. Related Data Structures	83
E-1. Action	83
E-2. Action Sequence	85
E-3. Association	87
E-4. Capability	88
E-5. Condition	89
E-6. Configuration Unit	90
E-7. Data Dictionary	93
E-8. Entity	96
E-9. Entity Type	97
E-10. Enumeration	99
E-11. Examination	103
E-12. Fidelity	107
E-13. Information	108
E-14. Interaction	109
E-15. Metadata	111
E-16. Network	114
E-17. Process Sequence	115
E-18. Reference	116

PRELIMINARY DRAFT COPY

E-19. State Transition	119
E-20. Task	121
E-21. Task Sequence	124
E-22. Use Case	126
E-23. Use Case Instance	129
E-24. Use Case Package	131
E-25. User	132
Index	135

PRELIMINARY DRAFT COPY

CMMS Representations

1 Purpose

The purpose of this paper is to propose a representation definition for Conceptual Models of the Mission Space (CMMS). Unfortunately, this area has many overloaded terms, and the usage prescribed here is to avoid overloading, meaning that only one of several meanings is included. One such term is CMMS. Herein, it is used to refer to the initiatives in this area of the Defense Modeling and Simulation Office (DMSO). Note that this is an attempt at defining the “logical” semantics of CMMS. It is important for the developers of CMMS and the style guide producers for configuration units to understand the semantics of CMMS; however, it is not necessary for users to have an explicit understanding of CMMS semantics as long as they understand the semantics implicit in the views of CMMS that they utilize. This is an attempt to describe requisite data to be stored; although representations are used to illustrate concepts, this is not intended to be a representational specification.

2 Executive Summary

This paper has been divided into several sections. This section describes each of the subsequent sections.

Part 3 provides an Overview. This section describes some of the more important CMMS concepts. Section 3.1 describes the purpose of CMMS. Section 3.2 outlines a conceptual process for producing CMMS. Section 3.3 covers major CMMS requirements.

Part 4 describes in more detail some important aspects of CMMS. Topics include descriptions of CMMS users, CMMS structure, a production scenario, and a detailed process for CMMS creation. This section concludes with discussion of model views and a conceptual model framework.

Part 5 describes the major components of CMMS and discusses how selected knowledge elements are ideally used. Part 5 uses a top-down address of the problem.

Appendix A provides a straightforward glossary (with annotation and examples, as deemed appropriate).

References are provided in Appendix B.

Appendix C discusses how physical environment, military environment, and civil environment are handled.

Appendix D gives entity examples.

Appendix E gives related data structures for certain concepts in CMMS.

PRELIMINARY DRAFT COPY

3 Overview

This section provides a high-level view of some of the important CMMS concepts.

3.1 What is the business of CMMS?

The mission of the military is to execute military missions. Mission is a highly overloaded word in the military context. The foregoing can be restated as follows. The principal objective of the military establishment is to be capable of successfully executing a wide range of military activities, not limited to warfare. The principal objective of CMMS is to serve as a repository for this information and to provide representations of this information that are useful to both military and non-military users.

What is the mission of CMMS? We do not believe it is to supply all information of interest to simulation developers. What part does CMMS provide? We believe that CMMS should provide descriptive information about the environment that the simulation addresses. Were we dealing with tactical systems, we could stop with providing the descriptive information in systems models, primarily with regard to interface definitions. Indeed, that is where CMMS stops; however, a major consideration of CMMS is that for simulations these descriptive models must also be implemented as a part of the system. It may be worthwhile to consider some types of information that CMMS is not intended to include. This information is of critical significance to the simulation director; however, the focus of CMMS is reusability, not universality. Examples of information not found in CMMS include the following:

1. implementation technology information, e.g., what platforms the simulation might execute on, what languages are used to write the software portions of the configuration, what operating systems or database management systems are to be used.
2. simulation execution specific information, e.g., orders of battle; mutable parameters, such as maximum engine thrust; network configurations.
3. prescriptive information, e.g., how should the simulation change the world around it, what problems is the simulation addressing, what are the hoped for benefits of the simulation.

3.2 How to do CMMS

Our discussion of CMMS will attempt to be top-down, breadth-first. Here we must demarcate what is included in CMMS and what is not. For example, an important aspect of knowledge acquisition is developing a focused context, which includes determining the relevant subdomains of military knowledge, determining the appropriate levels of fidelity and resolution, and prioritizing the efforts. Developing a focused context is essential to an effective effort; however, this is not addressed within CMMS support. The parts of the knowledge acquisition and engineering process which are supported by CMMS are the following.

PRELIMINARY DRAFT COPY

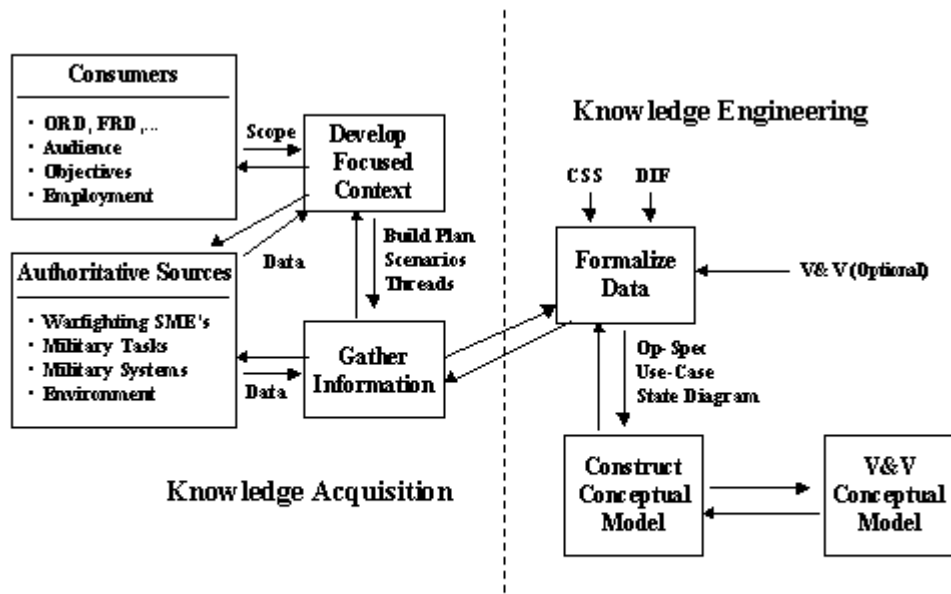


Figure 1. Describe Military Operations
(conceptual modeling)

Knowledge Collection/Gather Information – This covers a broad area and in some cases may include a sequence of activities. A convenient subdivision is into retail and wholesale. What we mean by retail knowledge collection (from the CMMS point-of-view) is the collection of information into CMMS at the retail level, in which we include, among other things, collection of knowledge from a single MOMS (Military operations mission space) SME (Subject Matter Expert)'s point of view or a JAD (Joint Application Development) or RAD (Rapid Application Development) workshop. Wholesale collection of knowledge is the acquisition in its native form of a body of knowledge initially collected, organized, and integrated outside of CMMS. Within retail there are still two types of collection. The first is fabrication, in which all knowledge must be entered; the second and preferable type is assembly, in which most knowledge is reused and minor modification and fabrication are necessary to support new uses and use cases. Note that CMMS does not require the use of these capabilities.

Knowledge Organization/Organize Data – This includes the syntactic aspects of compliance with CMMS. This is the conversion, for example, of a narrative into defined fields. It may be performed in real-time by a knowledge engineer, i.e., entered in a (somewhat) canonical form at the time of collection (SME recital, RAD workshop), or it may take place as a conversion from some non-CMMS representation. Note that the level of syntactic compliance may be variable (as opposed to binary). For example, all verbs and entities may be entered, but some relationships may be implicit (from definitions) as opposed to explicit within the appropriate data structure. The objective here is to maximize utility. The conflict is between a very rigorous definition, which we believe increases the

PRELIMINARY DRAFT COPY

utility of complying representations, and the non-utility of rejecting significant bodies of knowledge. The approach recommended here is to define a measure and report compliance with the absolute standard by the measure. If the best available is only 15% compliant, then that is what will likely be used. Conversely, given a choice between representations of 85% compliance and 15% compliance, all other issues equal, the 85% will be used most often. Indeed, there likely will be multiple objective measures. Subjective measures will be recorded through VV&A/C (Verification, Validation, and Accreditation/Certification) pedigrees.

Knowledge Modeling/Construct Conceptual Models – This is the step involved with translating non-object representations into conceptual object models. By conceptual object models (COMs), we mean first- or second-order abstractions of real world objects. Although other representations, for example, those oriented toward representing event flows, are useful and valid for some purposes, we believe construction of conceptual models is an essential ingredient of CMMS, in order that its reusability by simulation developers be maximized. It should be noted that although conceptual models certainly are related to implementation object models (IOMs) in that they form a partial specification of the behavior of the IOMs, there is no requirement for there being simple relationships (e.g., one-to-one) between COMs and IOMs. Indeed, there may conceivably be highly complex relationships (i.e., many-to-many) between COMs and IOMs. Note also that process-oriented or other non-object representations may be more meaningful to users, testers, and MOMS SMEs. Still another issue is that CMMS is directed toward knowledge of the military operations domain. A trainer is concerned not only with military operations that are the objective of the training exercise, but also the training process. How many trainers and students can be accommodated simultaneously is an issue of the training process but is unrelated to military operations, save as constraints on the exercise realism. Issues concerning the type of installation and computer platforms for which the system is developed are clearly implementation dependent and not a concern of CMMS. When this step takes place on a wholesale basis, adequate information to create conceptual object models may not be available. In this case, additional retail-level effort must take place before the model can be integrated. Measures of semantic compliance would be appropriately applied to the product of this step and the next.

Knowledge Integration/Expand Conceptual Models – This is the step involved with developing semantically meaningful relationships between organized knowledge components. A component might be a configuration unit. It may be combined with other components to provide even larger components; however, this requires that name space collisions be removed and appropriate relationships be developed between conceptual object models of the two precedent components.

We now look at ways of viewing conceptual models.

3.3 Objectives of CMMS

The objectives of CMMS are summarized as providing to users conceptual models that are correct, documented as bona fides, usable, consistent, composable, reusable, and produceable as simulations.

PRELIMINARY DRAFT COPY

3.3.1 Correctness of Conceptual Models.

CMMS should provide a correct representation of the military operations mission space. Correctness may be amplified in the following aspects.

1. The allowable syntax and semantics should be adequate to represent all activities of the military operations mission space.
2. Any population of data structures should be viewable in a form recognizable to its author(s).
3. Any population of data structures should be viewable in a form acceptable to examiners of the data set and to examiners of object systems.
4. To the extent that elements or structures are modified in CMMS, the modification should represent a correct translation.
5. Since one of the functions of CMMS is to provide information to users naive in the military operations mission space domain, it must provide information in a form which facilitates correct interpretation. Note that this objective implies many other objectives.

3.3.2 Documentation of Conceptual Models.

CMMS should provide adequate background information to enable a potential user to conduct a preliminary evaluation of suitability. Types of information included here are the following.

1. Production pedigrees provide information concerning production of the conceptual models. Information includes producer, production sponsor, dates of production, and data sources.
2. Examination pedigrees provide information concerning verification, validation, and certification of the data. Information includes examiner, examination sponsor, dates of examination, and reference materials, such as specifications and/or benchmarks.
3. Usage pedigrees provide information concerning access and utilization of the data. Information includes user, intended usage, and dates of access. Granularity of this data is determined individually by production sponsor (owner). One data set may collect no information concerning usage, while another collects everything at hand, including date/time of usage and amount transferred.
4. Integration pedigrees include date of attempted integration and exceptions noted in conversion (if applicable) and integration.

PRELIMINARY DRAFT COPY

3.3.3 Utility of Conceptual Models

The utility of information included in CMMS should be maximized. The principal means of maximizing its utility are the following.

1. “Tightly framing” CMMS with regard to its domain of application, i.e., the military operations mission space, will improve both its usability and the efficiency of the acquisition process. Examples of “tight framing” include the use of data dictionaries and SVD[PI]* (subject-verb-direct.object-preposition-indirect.object) to enforce common and constrained semantics and syntax, the use of common data structures such as EATI (Entities, Actions, Tasks, and Interactions), and the use of stereotypical (specific to military domain) associations and interactions. These models should be unambiguous in their representation within the context of military operations.
2. Engineering the conceptual models to the greatest extent possible without compromising reusability. In other words, engineer the knowledge until the point where further engineering would step into the realm of implementation assumptions and constraints.
3. Using consistent measures of resolution and fidelity. This issue of appropriate resolution is significant even within a project, as there may be needs for both variable resolution because of differing intended uses and consistent resolution within an intended use.
4. Use of framework models as discussed below.

3.3.4 Consistency of Conceptual Models

The objectives of consistency of effort and consistency of product are essential to realizing production efficiency and effectiveness. These key goals are facilitated by the following.

1. Using data standards to assure consistency in data elements.
2. Using common syntax and semantics to assure that military relevance is maintained.
3. Using constrained common syntax and semantics, including stereotypical elements, and knowledge structure to provide efficient production and extraction of CMMS knowledge.
4. Using style guides to assure that use of specific tools conforms to standard usage.

3.3.5 Composability of Conceptual Models

The composability of conceptual models is an objective because of its importance in achieving the following.

PRELIMINARY DRAFT COPY

1. Facilitating knowledge acquisition by disparate producers for the same development effort.
2. Facilitating construction of specifications using a building block approach.
3. Facilitating definition of different execution module compositions.

3.3.6 Reusability of Conceptual Models

Reusability of conceptual models is a key to efficient and effective production and usage of CMMS. It is important in the following areas.

1. Reusage between programs is essential for reducing long-term costs of military models and simulations.
2. Reusage between functions or phases of simulation development, e.g., between developers and VV&A, is essential to prevention of redundant efforts and inconsistency in the scope of knowledge used.
3. Reusage of conceptual model components facilitates reduced acquisition and engineering efforts with respect to related models.

Some of the ways that reusage is promoted are the following:

1. Data standards as discussed in 3.3.4 above.
2. Common syntax and semantics as discussed in 3.3.4 above.
3. Knowledge structure as discussed in 3.3.4 above.
4. Style guides as discussed in 3.3.4 above.
5. A data interchange format providing standardization to interoperability standards.
6. A conceptual model framework promoting usage of previously defined knowledge elements both for consistency of meaning and efficiency of creation. This is discussed in more detail in 4.7 below.

3.3.7 Produceability of Conceptual Models

That the conceptual models of CMMS be produceable as simulations is crucial. This goal involves the following.

1. The conceptual models should be designed for high efficiency. Means of maximizing efficiency are “tightly framing” CMMS (i.e., using data dictionaries and SVD[PI]* to enforce common and constrained semantics and syntax, using common data structures such as EATI (Entities, Actions, Tasks, and Interactions), and using stereotypical

PRELIMINARY DRAFT COPY

associations and interactions) and engineering the conceptual models to the greatest extent possible without compromising reusability as discussed in 3.3.3 above.

2. The conceptual models should be naturally translatable from the source documents.
3. The conceptual models should be composable as discussed in 3.3.5 above.
4. The conceptual models should be reusable as discussed in 3.3.6 above.

4 CMMS Description

4.1 Introduction

Throughout our description and discussion of CMMS, we should remember that this effort involves the intersection of three subject matter domains: the domain of military operations, the domain of knowledge representation, and the domain of software development. The areas of application of each of these domains are not trivial.

4.2 CMMS Users

Another complexity of this environment is the heterogeneity of the users. A principal input to the creation process must be the warfighter, or military operations mission space (MOMS) subject matter expert (SME). Two characteristics must be considered here. First, it is necessary that the MOMS SME (or appropriate doctrinal pronouncement) provide the inputs since no one else can speak authoritatively to behavior in the MOMS. Second, it is difficult to get the information necessary in most cases. Some SMEs may be able quickly and efficiently to break down the tasks; however, the nature of the SME is that he no longer works with atoms with respect to tasks or organization. His perception of the mission space is to categorize and summarize into complex abstractions. Where a software developer would see thousands of soldiers, a Brigade (BDE) commander (CDR) may see a few battalions. Where a software developer would see tens (or hundreds) of activities, the BDE CDR may see one task (perhaps partially complete). The BDE CDR knows that a battalion is composed of several hundred soldiers of various capabilities and specialties and that a multitude of activities must take place in order for a specific large-scale task to be completed; however, it is likely that this is not his working view of the problem. If it were, he might have difficulty functioning in real-time because he would be dealing with an overpowering amount of detail. He naturally makes an abstraction of units. For a BDE CDR this might be companies and batteries. For some commanders it might be platoons or battalions. When the software developer builds a simulation, he probably cannot use the command-level abstraction. His simulation requirements may require a great deal of mundane detail. At least one software developer must know each activity; otherwise, he won't be able to replicate real world tasks in the simulation he is to develop. Similarly, the tester needs to know that when the battalion moves from area 1 to area 2, all of a precisely defined set of soldiers and equipment must be moved. Even the trainer has a different view. Because of his focus on specific skills, he has to break things down into a detail fine enough to assure that the training will lead to acquisition of the appropriate skills by the trainees. This is important for two reasons: the trainer must understand the CMMS so that he can

PRELIMINARY DRAFT COPY

understand what are the training effects supported by the system, and the detail of the system and the CMMS must be adequate to support the requisite skill acquisition. From the descriptions at his disposal, plus his personal knowledge, he must determine the key skills for officers (and men) to acquire. These key skills may be beneath the higher echelon commander's perception. Again, he has the knowledge detail, but, in essence, what he deals with in the field is the knowledge summary. Even the officer who is performing the tasks may have developed important skills that he uses without thought. When asked a question, his immediate response will be extremely relevant to another experienced commander but will most likely not address the issues of the trainer, tester, or developer. Uncovering this level of "forgotten" detail will likely require iterative questioning and possibly different representations before the essential information is divulged. These differing representations will be described as views. Similar constraints upon system specification and implementation will come from other system end users, e.g., those concerned primarily with analysis of military operations or those concerned primarily with acquisition and development of military systems and equipment.

4.3 CMMS Knowledge Elements

The two principal subjects of discussion here will be views and knowledge elements. Knowledge Elements (KLs) are the components of CMMS that we believe are necessary and sufficient to a complete representation. However, they clearly do not represent the only complete representation. Other forms may use a different nomenclature, or even a different decomposition of the mission space, and arrive at different representation. It is hoped that such alternative representations will be bidirectionally translatable with this representation. Where we have represented data structures, we have attempted to provide a logical representation disregarding artifacts of a physical implementation such as surrogate keys. Views are important to the knowledge development process but do not require persistence. That is, they can be derived from the KLs. A different representation may define elements of our views as KLs, and some of our KLs may only be view elements; nevertheless, the same knowledge must be saved. Otherwise, one of the representations has unnecessary redundancy or is deficient in its knowledge structure. Redundancy is clearly the lesser sin. An in-depth technical discussion of specific KLs is provided in Section 5 below.

4.4 A Scenario for CMMS Production

An approach to this problem is to look at some major activities that the military might be expected to undertake. A scenario of current significance is the evacuation of peacekeeping troops. In such a scenario are thousands of men performing thousands of roles. How can a person grasp this, particularly, someone who is not familiar with this domain? One way is to start at the top. The President tells the Chairman of the Joint Chiefs of Staff (CJCS), "Evacuate them peacekeepers." This is perhaps an inadequate abstraction of the problem. The reality is that the "GO" command was preceded by feasibility and planning directives. Before the "GO" command is given, there has probably been some mobilization and redeployment of resources, including both men and materiel, as well as significant planning and wargaming of alternative courses of action and condition sets. We will assume that these issues are being addressed elsewhere. This assumption is the first example of paring the tree of activities. We will never be able to address thousands of activities, so how can we proceed? We proceed by using tree paring

PRELIMINARY DRAFT COPY

in two ways. One way is to subdivide the problem (divide a large problem into smaller ones and then conquer the smaller ones in detail). The other way is to discard limbs of the tree that are out of scope, substantially redundant with other limbs, or not significant for the problem being addressed. At this high level, we hope that paring will fall into the first category—divide and conquer.

What we have done so far is define an actor (President), a request (evacuate peacekeeping troops), and a role (CJCS). This is an example of a use case assignment—a use case has been assigned to CJCS by the President.

This representation is still a long way from the behavior we are interested in, so let us assume that the problem has been subdivided by phases and that we are only concerned with the operational phase—what happens after the task force is in situ and fully organized, e.g., Joint Operations Center (JOC) is set up. We can then change our view to actor (CJCS) requests (execute operational phase of peacekeeper evacuation) role (Commander, Joint Task Force—CJTF). This represents a significant paring of our tree hierarchy of activities. If necessary, we can return later to address other phases of our scenario.

The simple request by the CJCS drives a large number of activities by the CJTF, many of which are realized as requests of others. For example, operational situation development may be realized as role (CJTF) requests (maintain situational awareness) role (JOC). Closely linked to maintaining situational awareness are some other tasks, such as assess situation, develop courses of action (COA), etc. Another way of expressing this process is as a collaboration of the CJTF, J-3 (Joint Task Force Operations Officer), JOC, and CJTF's subordinate units performing operational situation development. Thus, one obtains a decomposition of Operational Situation Development into activities, such as maintain situational awareness, assess situation, develop courses of action, etc., as a collaboration of the CJTF, J-3, JOC, and subordinate units. Further decompositions may address portions of this collaboration. For example, a decomposition of “maintain situation awareness” may include generators and forwarders of situation events, as well as information on how events are selected for situation assessment.

Eventually, this process will reach the level of descriptions of how a watch officer assesses a situation. Where the line is crossed between requests between entities and description of activities performed by entities is significant. This is the line at which roles start to be filled in with characteristics and capabilities. In this discussion we make the distinction between actor and role. Actors are entities for which characteristics and capabilities are not filled in. Roles are entities for which the characteristics and capabilities necessary for one or more specified activities are filled in. A player is an entity that has been enhanced in two ways. First, basic behaviors, such as survivability, have been added, and second, multiple roles are assigned to fill the entity out. Although actors, roles, and players are all entities, from the perspective of a specific mission definition actors may be viewed as outline figures, roles as filled in figures, and players as three-dimensional figures.

A player specification is the product that will cause an object-oriented programmer to go forth and be fruitful (produce lots of objects). It represents a direct composition of roles and indirectly a composition of task descriptions. In effect, the roles become handles for quick composition of players from roles.

PRELIMINARY DRAFT COPY

Figure 2 lays out the components resulting from following the plan of attack described above. Specifying use case collaborations of successively greater refinement results in atomized requests—requests that are clearly allocable to a single agent or entity of interest. These requests are then mirrored as operations of a role (or possibly a player). They may be decomposed and then elaborated, or they may be elaborated, or they may be elaborated and decomposed, depending on the objectives of the CMMS production and what appears to be appropriate for the operations and roles in question. The operation description shown in Figure 2 is of appropriate resolution but should still be enhanced with more state and event data, such as how forces, support, and technology availability are determined and what they are benchmarked against to determine feasibility.

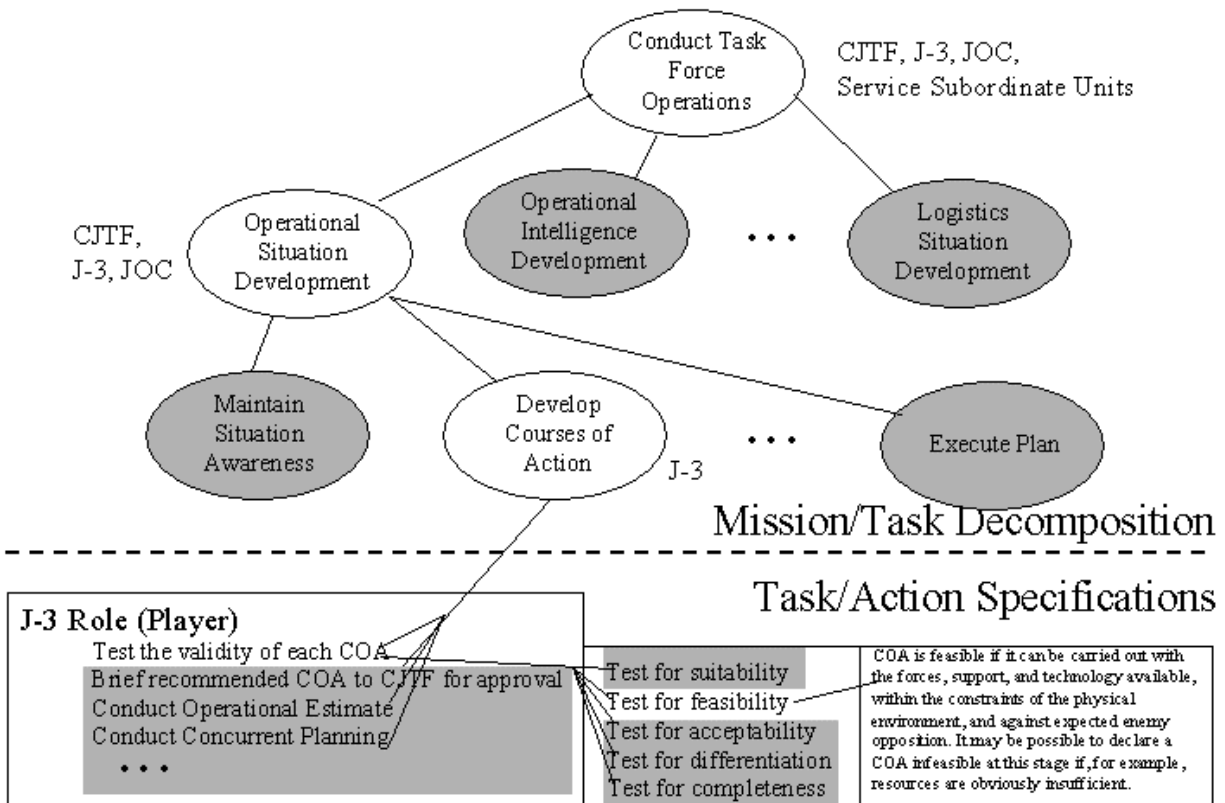


Figure 2. Mission Decomposition/Action Specification

4.5 A Process for CMMS

We can now lay out a formal process for producing the player specifications that the object-oriented programmer just can't wait to get. It is of note that the stages of knowledge acquisition and knowledge engineering take place concurrently and that there are three possibilities for knowledge acquisition sources. One may collect knowledge from military doctrine documents, from knowledge-acquisition documents, or from SMEs. The principal steps are the following.

PRELIMINARY DRAFT COPY

- (1) Determine use cases and players of interest (and scheduling priorities).
- (2) Acquire knowledge from military documents, knowledge-acquisition documents, or SMEs.
- (3) For each use case:
 - (a) Start with a use case definition (Actor-requests-agent).
 - (b) Decompose use case into processes and threads (composed of actors, agents and requests).
 - (c) Continue decomposition until one reaches simple agent-request relationships.
 - (d) When requests are simple, specify agent operations as homogeneous activities with simple transitions and conditions.
 - (e) If appropriate, provide state transitions for agents:
 - (i) Agent has significant executional dependencies upon state, and
 - (ii) State space of agent is non-trivial.
- (4) For players of interest:
 - (a) Identify player.
 - (b) Generalize (add in) basic behaviors.
 - (c) Generalize (add in) appropriate roles.
 - (d) If appropriate, provide state transitions for players:
 - (i) Player has significant executional dependencies upon state, and
 - (ii) State space of player is non-trivial.
- (5) For roles essential but not handled above:
 - (a) Identify additional players required.
 - (b) Allocate basic behaviors and essential roles, as appropriate.
- (6) Verify and validate operational specifications and interactions of players.
- (7) Integrate:
 - (a) Convert non-standard identifiers to standard.

PRELIMINARY DRAFT COPY

- (b) Verify that inputs and outputs match.
- (c) Substitute higher-resolution players for lower-resolution players.
- (d) Verify that all of the above interacts appropriately.
- (8) Validate integrated players and interactions:
 - (a) Are process aggregations appropriate?
 - (b) Are use case traces appropriate?
 - (c) Etc.
- (9) Feed to Object Oriented Programmers:
 - (a) Use case package players must be integrated with user space players.
 - (b) Both sets of players must be integrated into execution and technological frameworks.
 - (c) Then, code.

Note that steps 1, 7, 8, and 9 are not part of CMMS but are included for context. Note also that although an object-oriented development process was posited, that is not a requirement in order to use CMMS.

4.6 Three Model Views

Three important views have been identified of models. In OOM&D (*Object-Oriented Modeling and Design*) [1], Rumbaugh describes Object, Dynamic, and Functional models. An object model provides the framework for the other two models. This is where operations and attributes are attached to classes. The dynamic model describes time and sequences. This is the model directed toward representations of events and states. The functional model is concerned with transformations of values, with what a system does, and with inputs and outputs. We could characterize these concepts as structure, control, and flow, respectively. For example, an association between two objects is a structure; the association may imply control or flow. An interaction is a flow between two objects; it may imply structure or control. A state may lead to an activity that causes a new state that enables a new activity. This impacts the timing of occurrences within structures and of flows. We would like to construct three-dimensional object models, models that have existence as structures, controls and flows. If a representation cannot support meaningful views in the three dimensions of structure, control, and flow, it is not complete. When we have finished our discussion, we will apply this test to assess the completeness of our representation.

Whether there should be three models or eight models is probably not an issue in itself. What is significant is how much the cost of additional representations is and whether or not all significant aspects of the conceptual object are covered in a view or model. The significance of

PRELIMINARY DRAFT COPY

aspects and the significance of costs are program-dependent; nevertheless, we need to consider the universe in determining what can be potentially included in CMMS. Programs will then decide what portion of the universe will be actually included in CMMS through their efforts.

4.7 Conceptual Model Framework

Great care must be given here so as to support exploitation of the full power of object technology in order to achieve maximal reusability while providing straightforward representations comprehensible by domain experts who may not be technically oriented. A deep inheritance hierarchy may be essential to mobilization of an effective implementation effort; this is not an appropriate view for those who are unfamiliar with inheritance concepts such as virtual classes and polymorphism. One approach is to allow some of the benefits of inheritance while avoiding those where errors in understanding are more likely to occur, e.g., to allow multiple inheritance but not to allow overloading of operation and task names. Thus, problems with an

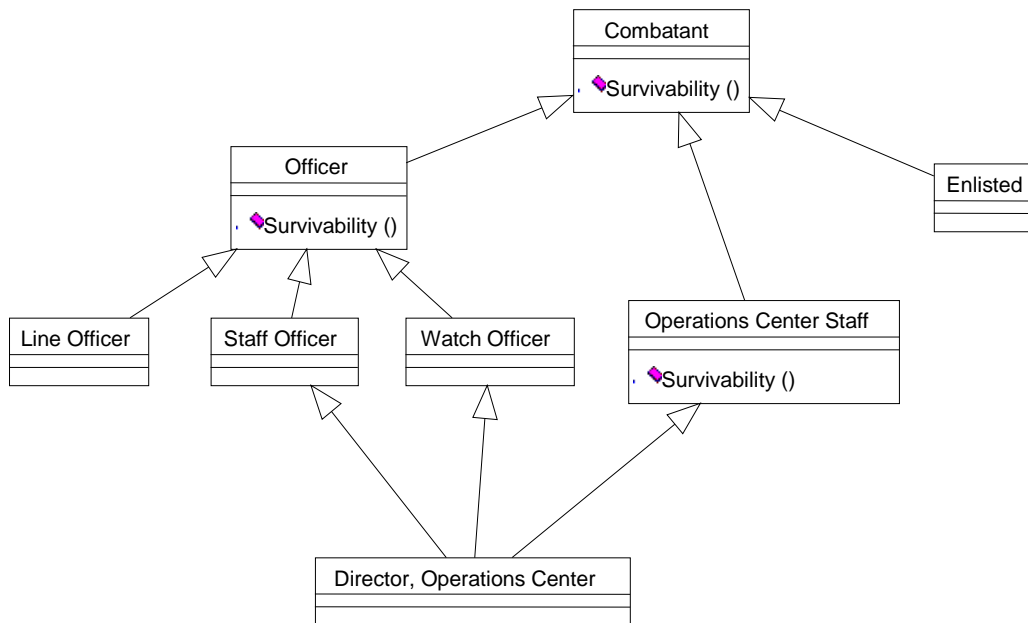


Figure 3. Multiple Inheritance Example

operation changing because of the insertion of a modified operation in an intervening class are avoided.

Figure 3 provides a graphic representation of the problem. The issue is which Survivability operation the Director of the Operations Center will use. Rules for handling the Combatant:Survivability are well established, and it is not really a candidate. However,

PRELIMINARY DRAFT COPY

resolution between Officer:Survivability and Operations_Center_Staff:Survivability is more complex; OO (object-oriented) conventions to resolve this conflict do not exist. We propose to follow another approach, which is not to allow such conflicts. An example of this approach is shown in Figure 4. The Figure 4 modifications assume that originally survivability was defined for combatants. Then it was determined that it was really different for officers. This gave rise to a problem for those elements, such as an operations center staff, who might be both officers and enlisted. The solution therefore was to rationalize the inheritance hierarchy as illustrated in Figure 4.

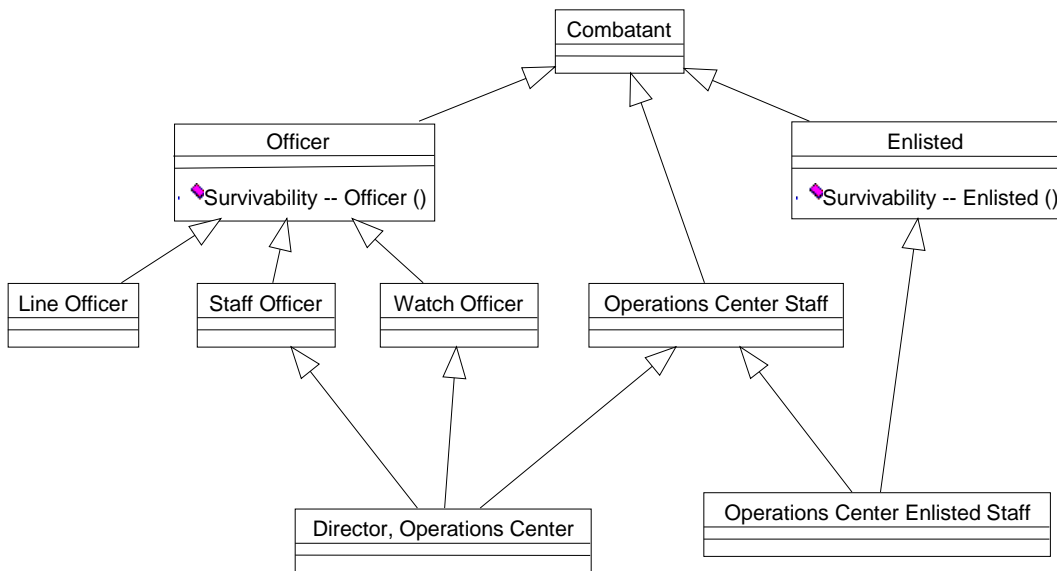


Figure 4. Modified Multiple Inheritance Example

5 CMMS Knowledge Elements and Discussion

We now begin the process of analyzing CMMS semantics, starting with high-level definitions and moving toward successively more detailed levels.

It is the goal that CMMS provide a traceable model. We use the term “traceable” here not in the sense that it takes on in object-oriented analysis in which traceability means that one can

PRELIMINARY DRAFT COPY

trace an analysis object or class, along with its attributes and operations, to the corresponding object or class in the design models and program code. Instead, we use the term here to mean that information in the model should be traceable to its authority. If one finds information, one ought to be able to find what authority applied to it. Traceability thus has to do with navigability in the model.

5.1 Use Case

Use Case:

Discussion – A Use Case is a coherent unit of functionality provided by one or more Objects. In the systems analysis arena, a Use Case represents a use of the system being developed by a beneficiary called an actor. In the military operations arena, a different definition is required. At one level, a Use Case is a stimulus of an Object that causes the Object to engage in behavior. This is the equipment to equipment type of Use Case. At another level, a Use Case is an assignment of a set of objectives to an Object, resulting in objective-oriented behavior by the Object. This is a mission assignment within the DoD hierarchy type of Use Case. Other examples include the response of a sensor or an operational unit to opposing force activity. For CMMS, we define a Use Case as a description of the behavior of an Entity (responding Entity) as the result of stimulation by another Entity (stimulating Entity); the behavioral description is limited to the externally observable aspects of the Entity's behavior.

We define Use Case from two points of view. The first part of the following definition is conceptual, and the second part is constructive or technical.

Definition – (a) *A Use Case is a description of the behavior of an Entity (responding Entity) as the result of stimulation by another Entity (stimulating Entity).* (b) *A Use Case is a coherent unit of functionality provided by one or more Objects.*

A Use Case is a configuration unit, meaning that the producing sponsor has ownership rights. It is optionally decomposable into subunits and may be associated with one or more task sequences. At the higher levels, it may be thought of as an organizational mission, conceivably as abstract as “military defend democracy.” At the lower levels, it tends to be much more concrete, such as “Mechanized Infantry Battalion attack heavily defended town.” Although a Use Case may have significant associated reference materials, the principal purpose of the Use Case structure in CMMS is to provide a nominal description of a moderate to high level military activity, to identify the principal participating entities, to identify significant relationships among participating entities, to identify conditions affecting the performance of the Use Case, and to identify objectives and measures of performance.

A Use Case is used to describe behavior at a high level. The principal artifact of description is a collaboration of two or more entities and the principal task requests passed between the entities. Messages are assigned sequence numbers. Equivalent message numbers indicate concurrency. A task request is a request from the sender for the performance of a task by the receiver. A Use Case is characterized by supporting

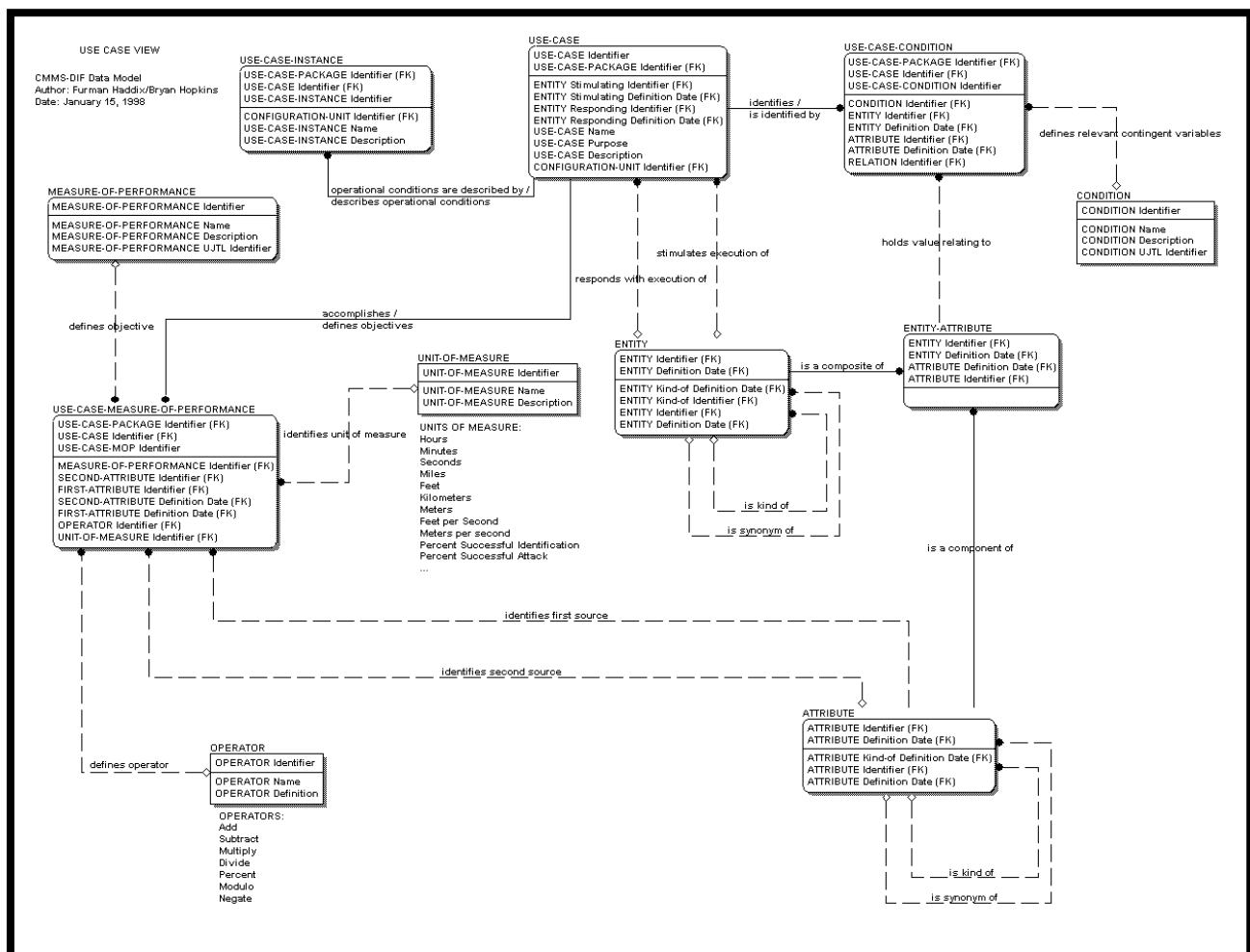
PRELIMINARY DRAFT COPY

documents, objectives, and measures of performance (MOPs). An associated Use Case Instance can provide additional information concerning the existence of conditions under which the Use Case might be executed and associated standards of performance. If specified, a Use Case Instance may be characterized by the following: order of battle; conditions of the physical, military, and civil environments; and standards for MOP achievement. Normally, if conditions are specified, it is because they are relevant to Use Case execution.

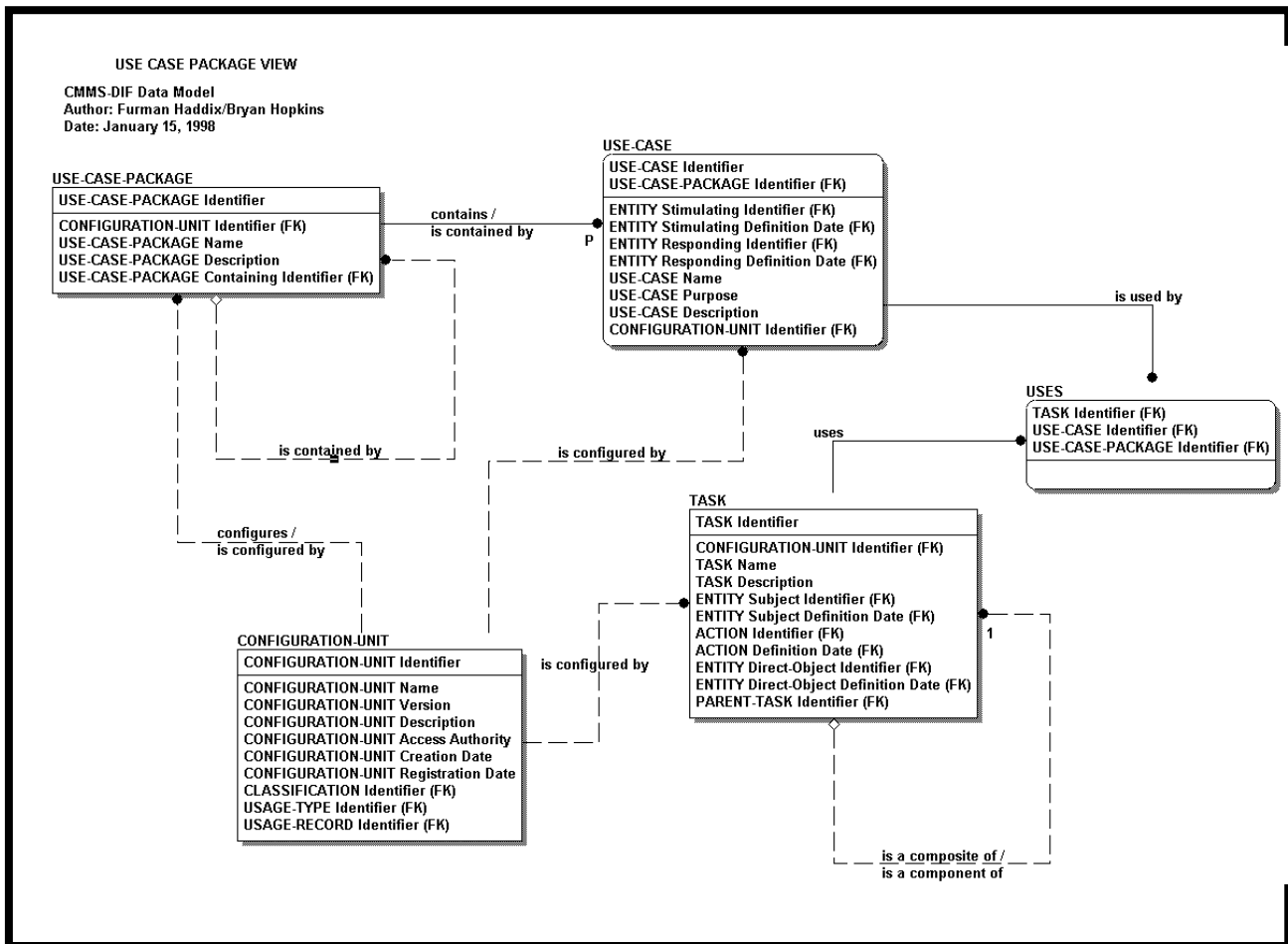
Use Case Package:

Definition – A Use Case Package is a set of Use Cases that share a common organizing principle, purpose, or feature. The Use Cases in a Package may be related by function, type, objective, performer, or other relationships.

The following Use Case View illustrates Use Case usage, and the following Use Case Package View illustrates Use Case Package usage.



PRELIMINARY DRAFT COPY



5.2 Action

Action:

Discussion - An Action in CMMS is an activity of the military operations mission space (MOMS). It is used as a template for construction of tasks. An Action would attempt to be as general as possible in specification of subject and direct and indirect object entities. For example, highly general entities would be entity, agent, nonagent, organization, or individual. A task would be much more specific and would have sources and sinks for its inputs and outputs specified. Both, however, would be expected to follow an SVD[PI]*-based syntax, and both would use terms based on CMMS data dictionaries. The SVD[PI]* format indicates that an Action description consists of a Subject entity, a Verb, a Direct object entity, and zero or more modifier phrases which include a Preposition and an Indirect object entity. An alternate form consists of a Subject entity, a Verb, and a dependent clause – SV[SV]D[PI]*.

We could use something like the following definition. “Action is a Task-template description having a single context-free meaning and consisting of a verb and optionally

PRELIMINARY DRAFT COPY

other semantic modifiers, i.e., subject, direct object, or indirect object phrases. The meaning of the Action is tied to a single meaning of its verb if the verb has multiple meanings.” This accomplishes our semantic objective; however, it is a syntactic nightmare. A possibly superior solution is to say that Action must conform to the SVD[PI]* syntax. The semantics are equivalent if we allow the use of agent as a generic subject and entity as a generic object. This means that a task is composed of an Action, a (possibly null) decomposition hierarchy, and (possibly null) conditionals. We define Action from two points of view. The first part of the following definition is conceptual, and the second part is constructive or technical.

Definition – (a) An Action is the alteration or transformation by natural force or human agency that produces an event such as move, sense, communicate, engage, or replenish. (b) An Action is a Task-template description having a single context-free meaning and consisting of a verb, subject, direct object, and optionally, indirect object phrases. The meaning of the Action is tied to a single meaning of its verb if the verb has multiple meanings.

Process:

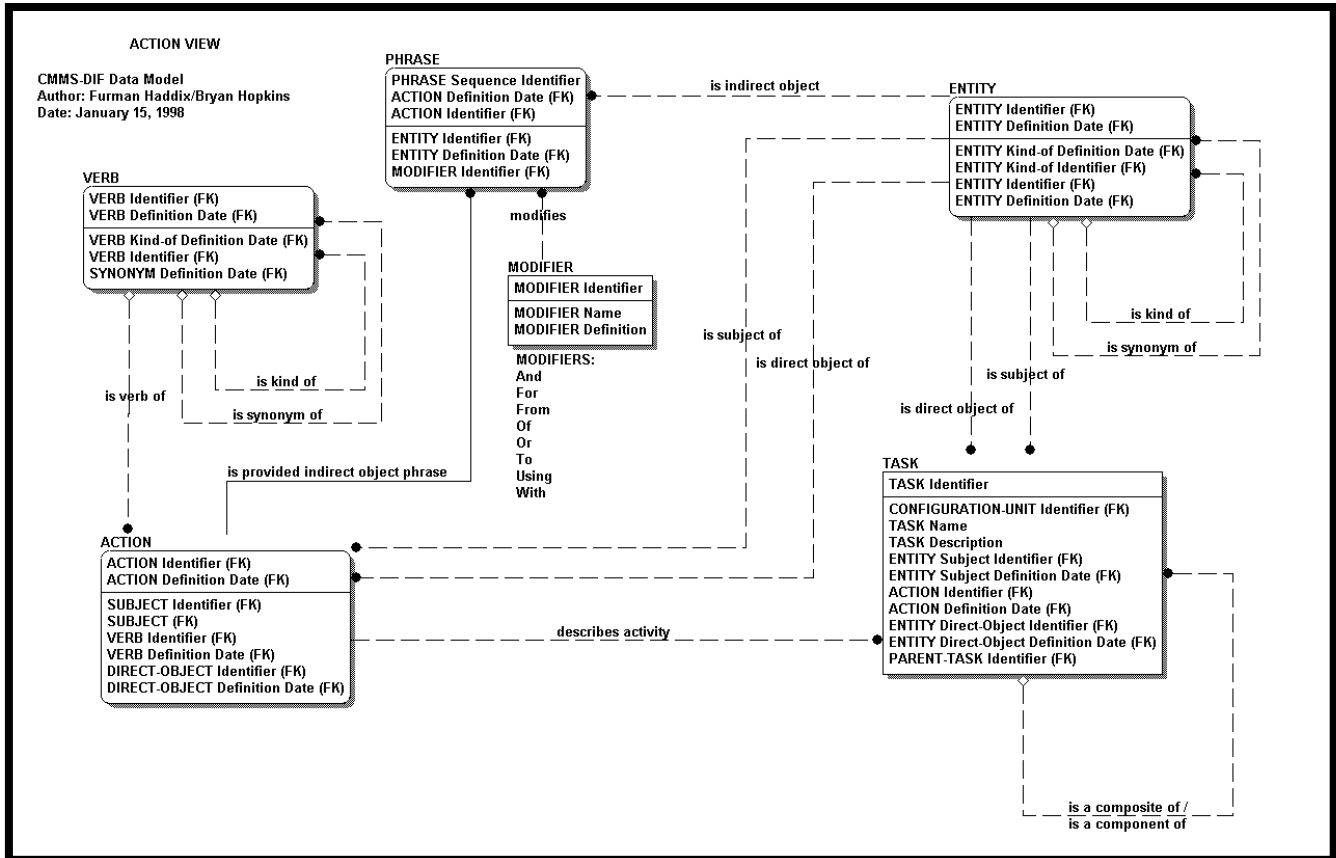
Discussion – A Process is a sequence of Actions in the same way that a Use Case is a sequence of Tasks. Furthermore, just as an Action is realized as a Task, so a Process is realized as a Use Case.

Definition – A Process is a sequence of Actions.

The following Action View illustrates Action usage

.

PRELIMINARY DRAFT COPY



5.3 Task

Task:

Discussion – A task is a purposeful behavior engaged in by an entity in order to satisfy a military objective. It is an activity of the military operations mission space. Minimally, it is associated with a subject entity, verb, and direct object entity. Common syntax and semantics are enforced in its name and its description. It is closely associated with other semantic elements. An action provides a template for one or more tasks. Hence, a task could be characterized as an instance of an action. Tasks could also be characterized as specializations of actions in that the entities associated with an actions are often generalizations of the entities associated with a task.

There are three levels of task specification.

1. A task request is normally a task name without a specification.
2. A task interface specification describes the inputs and outputs associated with the task.

PRELIMINARY DRAFT COPY

3. A task operation specification describes the behavior in performing the task, in addition to the interface information.

A task follows an action template. A task may also be represented as a use case. An action is a composite of a verb and several entities. Syntactically, an action is a transitive verb and at least two entities if it has an SVD[PI]* structure [3]. SVD[PI]* is translated into Subject-Verb-Direct.object-Preposition-Indirect.object. A subject is a type of agent entity; a direct object can be any entity, as can an indirect object. The PI should be interpreted as an indirect object phrase. The indirect object “helper” need not be strictly limited to prepositions. An interesting semantic question is how many phrases should be allowed per action. For example, we would like to represent concepts such as the following: Agent crosses river using rubber boats with supporting fire from supporting units. An alternative would be to represent this as two tasks. The “with supporting fire from supporting units” is then considered an action of another subject. The original sentence may be represented as follows:

Subject: agent

Verb: cross

Direct Object: river

Indirect Object Phrase:

Indirect Object: rubber boats

Phrase Modifier: using

Indirect Object Phrase:

Indirect Object: supporting fire

Phrase Modifier: with

Indirect Object Phrase:

Indirect Object: supporting units

Phrase Modifier: from

This problem is compounded by the possibility of conjunctive and disjunctive phrases, for example: Agent crosses river with supporting fire from supporting artillery and armor units, or agent crosses river with supporting fire from supporting artillery or armor units. This implies sequencing and typing. The sequencing appears to be essential not only to identify compound phrases but because the order affects the semantics of the action. This has progressed a long way from verb. In other words, the disambiguation of verb should be a separate step. This concept can be represented as action so that the action becomes the disambiguated verb. Although the SVD[PI]* structure seemingly provides overkill, it is usable as a structure for actions and a basis for task. Note that the action entities are not necessarily the entities of the task, as the latter may be more specific.

PRELIMINARY DRAFT COPY

One of the more difficult issues is how to obtain reusability of tasks. Some elements of tasks apparently can be reused only by creating additional instances. Action templates appear to be reused by reference since actions are data dictionary elements. Additional task instance creation implies a new Configuration Unit. We hope that the concept of generic tasks and use cases will provide an adequate capability for high-level reusability with the capability of creating new specific or generic use case instances for modification. If we consider the generic use case a template, then we can build new templates from the old. We can also reuse the old generic template to build new operations by using new use case instances. Use case instances too can be reused with different generic use cases to form new operations or copied to assist construction of a new use case instance.

Still another issue with tasks is decomposition. An important aspect of tasks is sequencing. An argument could be made that sequencing is due to dependencies and dependencies are specified in relationships, and that might well be valid if it were not for the significance of actors. The problem is that the activities of actors are unspecified. So dependencies involving actors may be underspecified or unspecified, and sequencing information may exist but have no explicit representation. It appears to be better to include explicit sequencing and reduce the risk of information loss at the cost of possible redundancy in information. Decomposition can be simply represented as a parent/child relationship. It is the sequencing that is an issue, and there are two aspects of this problem: 1) if the parent is subtask 2, what are the sequence determinants of the children; 2) what sequencing relationships should be shown – concurrent and serial seem obvious, but what others are essential? Note that this is an issue for all inter-task relationships. The answer we have chosen is to say that sequencing, including iteration, can be defined as a conditional, and that conditionals and relationships are explicitly applied to the steps – the bottom elements of the task decomposition tree – and are implicit attributes of all other tasks.

Conditionals represent another challenge with respect to task representations. This is an important part of any military task specification. Most representations address at least preconditions and postconditions. We have identified three types of contingencies – IF, WHEN, and WHILE. IF is an instantaneous, non-blocking contingency. IF means that if the predicate is true, behavior is this, else do another behavior. WHEN is an instantaneous, blocking contingency. WHEN means that you will do this; however, you don't do it until the predicate evaluates to true. WHILE means as long as this predicate is true, do (until task completion), else do another activity (until task completion). The activities of interest under these controls would be START, PAUSE, CONTINUE, and TERMINATE. START is used with regard to a specific TASK, which may be the instant task. PAUSE, CONTINUE, and TERMINATE apply to the instant task. It should be pointed out that these conditionals encompass both UJTL (*Universal Joint Task List*) [4] conditions and flow of control conditions.

The first part of the following definition is conceptual, and the second part is constructive or technical.

Definition – (a) *A task is a description of a military activity, including the activity performer and the activity object(s). It may be decomposed into subtasks (recursively decomposable) or steps (atomic).* (b) *A task is the execution of an action by an entity. The*

PRELIMINARY DRAFT COPY

entity initiates execution when specific entrance criteria are met. During execution the task may receive or consume one or more inputs from suppliers, may produce or deliver one or more outputs to receivers, and may change one or more entity attributes. Execution continues until specific exit criteria are satisfied.

Generic Task:

Discussion – We define two types of task: generic and specific. Generic uses types of entities (which are also entities); specific uses entities that are not types or at least low-level types. Either type of task is a valid CMMS specification; only the specificity is different.

Definition – *A generic task is a task in which participants have been identified by entity type.*

Specific Task:

Definition – *A specific task is a task in which participants have been identified by specific entity.*

Task Step:

Discussion – A task may be decomposed into subtasks or task steps. A task step is an atomic component of a task. It may not be broken down into further subtasks or task steps. A task step must share the same use case with the task of which it is a component.

Definition – *A task step is an atomic component of a task.*

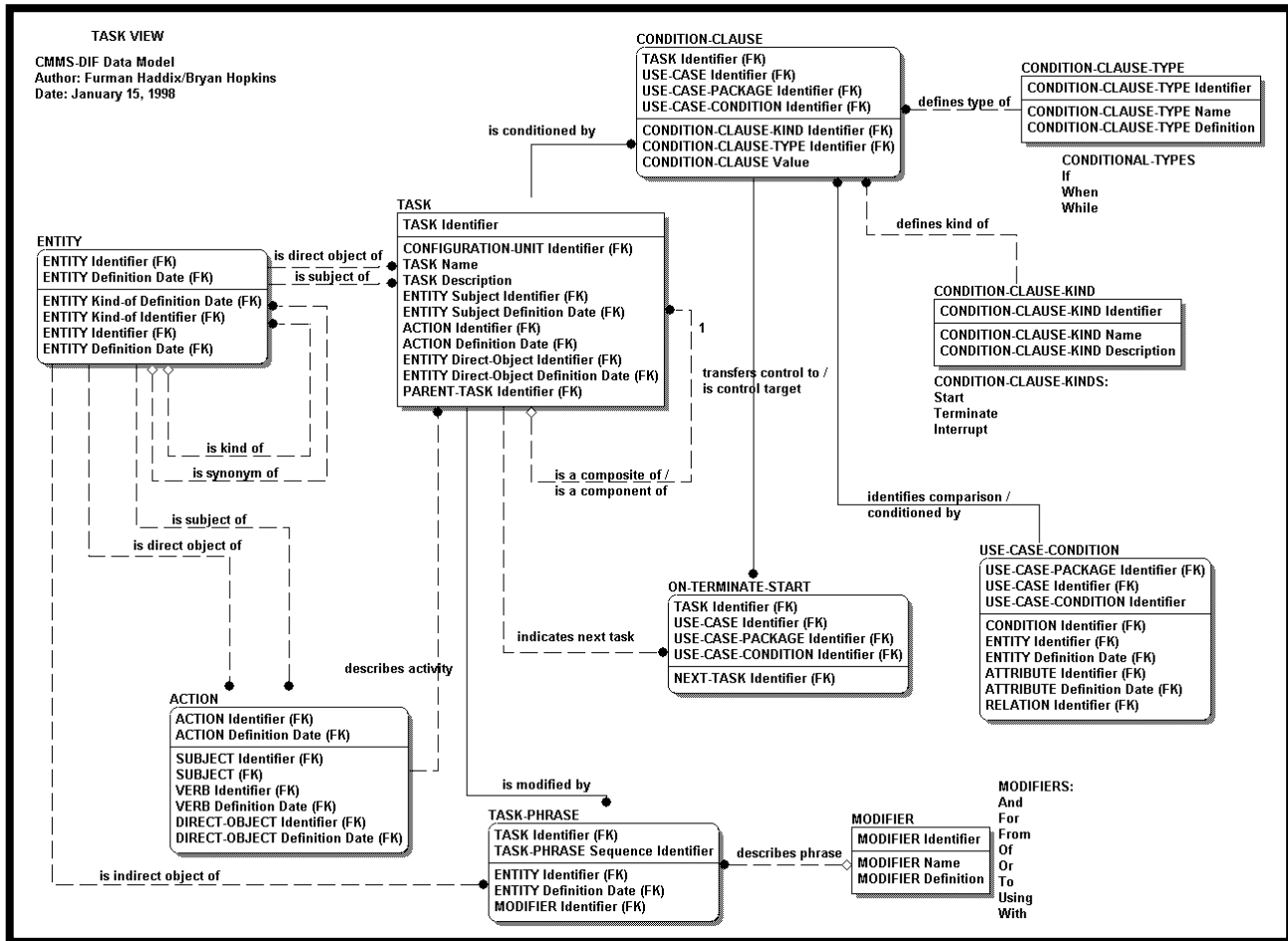
Subtask:

Discussion – A subtask, like a task step, is a component of a task. However, a subtask is itself recursively decomposable; it may be broken down into further subtasks or task steps. A subtask must share the same use case with the task of which it is a component.

Definition – *A subtask is a recursively decomposable component of a task.*

The following Task View illustrates Task usage.

PRELIMINARY DRAFT COPY



5.3.1 Nominal Tasks

Nominal tasks are used in articulating relationships between entities within the context of a use case. They are called nominal because although they are named tasks, they are normally not filled in with detailed specification information. Nominal tasks are used where context and explanatory information is desirable but the effort to complete a substantive task definition is not warranted. In a use case/task decomposition hierarchy, nominal tasks would normally be found in the upper and intermediate parts of the hierarchy, while substantive tasks would be used in the lower parts.

5.3.2 Substantive Tasks

Substantive tasks are used in describing work performed by agents. They differ from nominal tasks in that they are filled in with detailed specification information concerning preconditions, postconditions, semantics, and substantive interactions with other tasks.

PRELIMINARY DRAFT COPY

5.4 Sequence

Sequence:

Discussion – One of the challenges of any large-scale simulation is to take large numbers of simulated entities and make them perform in concert. These activities may be referred to as processes (because of the multiple flow control threads), tasks (because this is a commonly used military term), or use cases. In CMMS, we have chosen to refer to the small-scale activities as tasks and the large scale as use cases. However, within a use case tasks cannot ordinarily be performed in an arbitrary sequence. Although inter-task interactions may provide some sequencing control, exclusive use of this medium would at best result in the creation of artificial synchronization tasks. At worst, it just would not work. Therefore, CMMS follows the principle that when there are tasks that need to be performed at the same time or at precise relative times, these constraints on execution should be explicitly stated.

A task sequence is a set of tasks having implicit or explicit sequencing relationships. Since task sequencing is achieved by specification of constraints on concurrent execution, the default is that all tasks in a sequence set can begin concurrently and end asynchronously. A task sequence by itself does not fully define an execution sequence since an operation specification also imposes constraints on task execution, e.g., conditional task execution logic is provided in the operation specification, and the absence of required inputs may defer or proscribe execution of a task.

Task sequencing works as follows. A set of tasks is identified to a task sequence. If no constraints are specified, all tasks may be executed concurrently. Constraints are of the form Task A Constraint Type, Task A, Task B Constraint Type, Task B. The identification of a task includes the entity performing the task, the task sequence identifier, and the task identifier. Task A constraint types are Begin(ning of task), and End (of task). Task B constraint types are Prior (to beginning), Begin, During, End, and After (end of task). UML (Unified Modeling Language) provides an incomplete sequencing in Collaboration Diagrams; annotation is required to provide a complete definition.

As stated below in the Interaction subsection, the synchronization of a transfer is dependent upon task sequence. The following examples of task sequencing also specify the transfer synchronization. The specification for each constraint should be interpreted as representing a single point in time. For example, “Begin – Task A; Begin – Task B” means that the point in time at which Task A begins is the same point in time at which Task B begins. “Begin – Task A; Prior – Task B” means that the point in time at which Task A begins is a point prior to Task B. “End – Task A; After – Task B” means that the point in time at which Task A ends is a point after Task B.

1. Concurrent execution:

Constraint 1: Begin – Task A; Begin – Task B.

Constraint 2: End – Task A; End – Task B.

PRELIMINARY DRAFT COPY

Transfers: Transfers are sent during Task A execution and received during Task B execution.

Constraint 1 forces Tasks A and B to begin simultaneously; Constraint 2 forces them to end simultaneously. This has the effect of not allowing Task A to conclude until postconditions for Task B are satisfied. The semantics of During depend on the cardinality of the transfer entity: $1 \rightarrow \text{midpoint}$; $2 \rightarrow 1/4; 3/4$; $3 \rightarrow 1/6; 1/2; 5/6$; etc. In general, $n \rightarrow 1/2n; 3/2n; 5/2n; 7/2n; \dots 1-(1/2n)$. Note that constraint 1 is not necessary if it is not important that A and B begin at the same time.

2. Sequential:

Constraint: End – Task A; Begin – Task B.

Transfers: Any transfers between A and B take place at the sole point of synchronization: End of Task A/Beginning of Task B.

3. Nested:

Constraint 1: Begin – Task A; Prior – Task B.

Constraint 2: End – Task A; After – Task B.

Transfers: Transfers are sent and received during Task B execution.

This forces Task B to be nested within the limits of Task A execution. The delay between the beginning of Tasks A and B can be momentary, as can the delay between the end of Tasks B and A.

4. Disjoint:

Constraint: End – Task A; Prior – Task B.

Transfers: Transfers are sent asynchronously at the end of Task A and received at the beginning of Task B.

The delay between the end of Task A and the beginning of Task B can be momentary. If a synchronous transfer is desired, the sequential constraint above can be used.

5. Unrelated:

Two tasks are included in the same sequence set but with no sequencing constraints.

6. Iteration:

Iteration is a repetition of either a single task or a task sequence. The sequencing constraints used to specify the other forms of task sequence are not relevant to specify iteration. The WHILE conditional discussed in section 5.3, “Task,” is an example of iteration. WHILE means that as long as the predicate is true, do the task until task

PRELIMINARY DRAFT COPY

completion, else do another activity until task completion. However, the WHILE loop is not the only type of iteration that may be relevant. For example, the IF conditional, which means that if the predicate is true, then the behavior is this, else do another behavior, is adequate to cause iteration if it has the appropriate predicate and is put in the right place, e.g., Do Task One, {Activities of Task One, IF Enemy still standing THEN START Task One}.

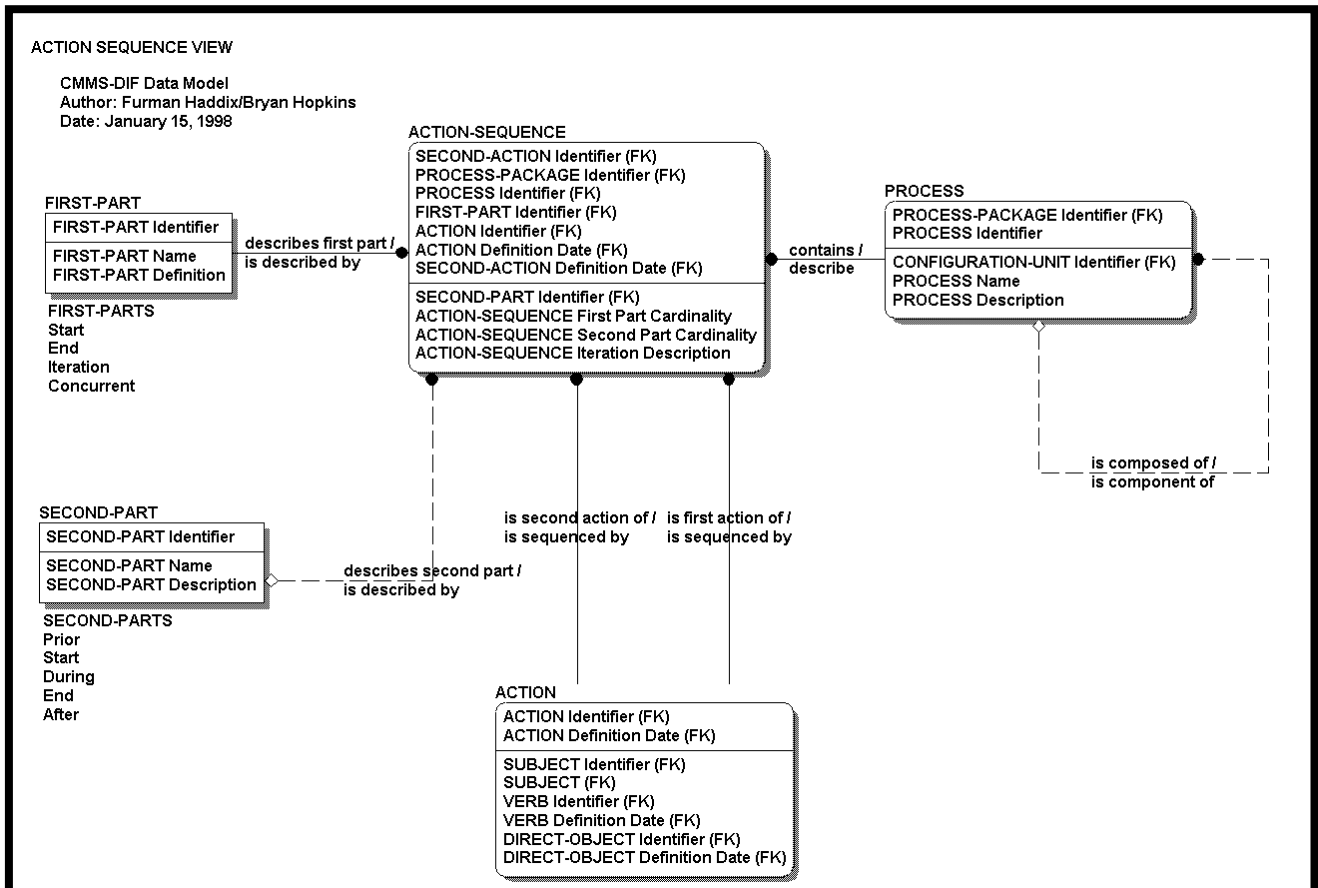
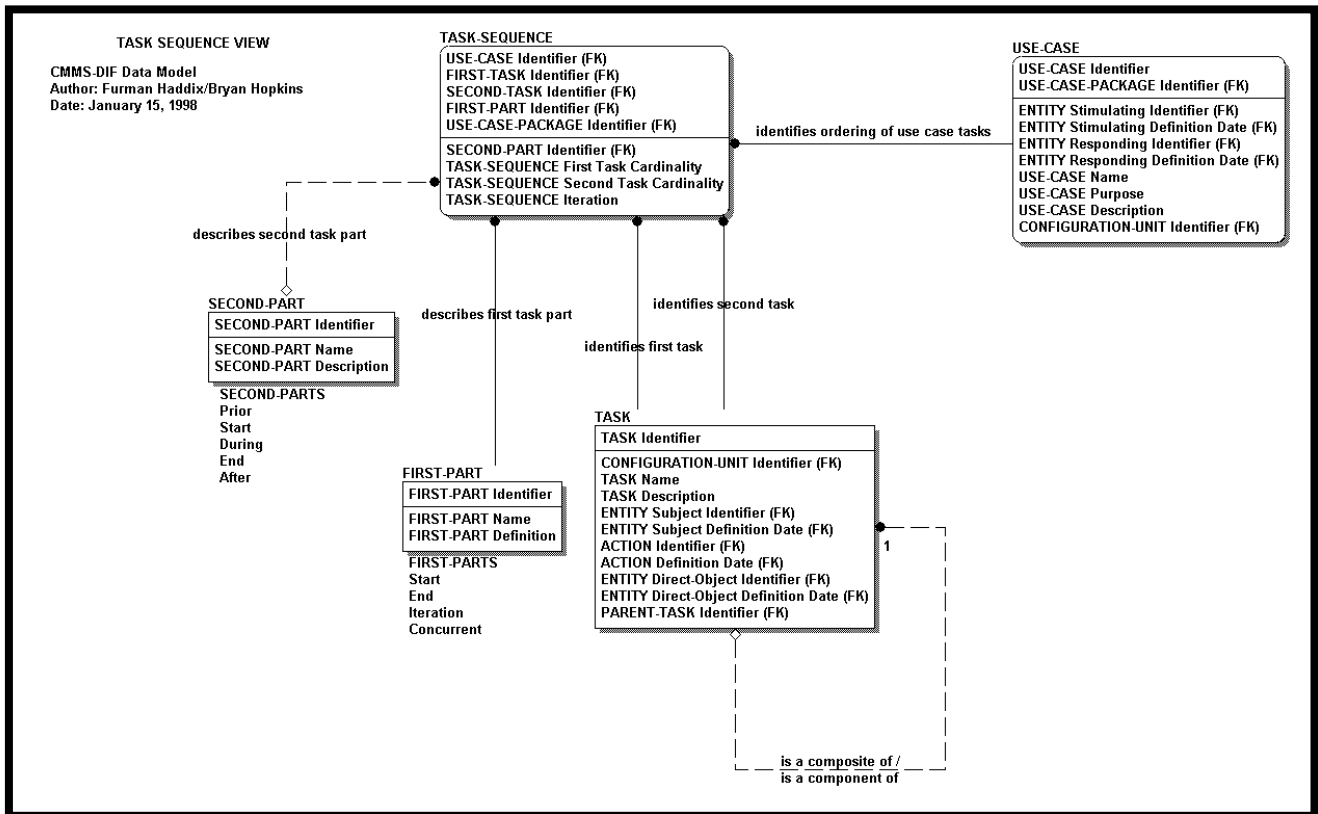
As discussed above, it is not clear that sequence information is always needed in addition to other interactions; however, it appears that is needed in some cases, and so the capability should be provided. Some sequencing information is included in conditionals. Since sequence is an optional specification, an adequate capability is provided for all foreseeable usage. A sequence is directional. It provides two types of specification – the relationship of the first task to the beginning of the second task and the relationship of the first task to the end of the second task. Thus, potentially, there would be two sequence relationships between two tasks. This likelihood appears to be very low. The relationship would thus be characterized as task 1 preceding, beginning coincidental, during, ending coincidental, and succeeding with respect to either task 2 beginning or task 2 end.

CMMS also defines sequencing for Actions and Processes. Sequencing for Actions and Processes works through constraints in the same way that sequencing for Tasks works, and the entire discussion above applies to Action and Process Sequence.

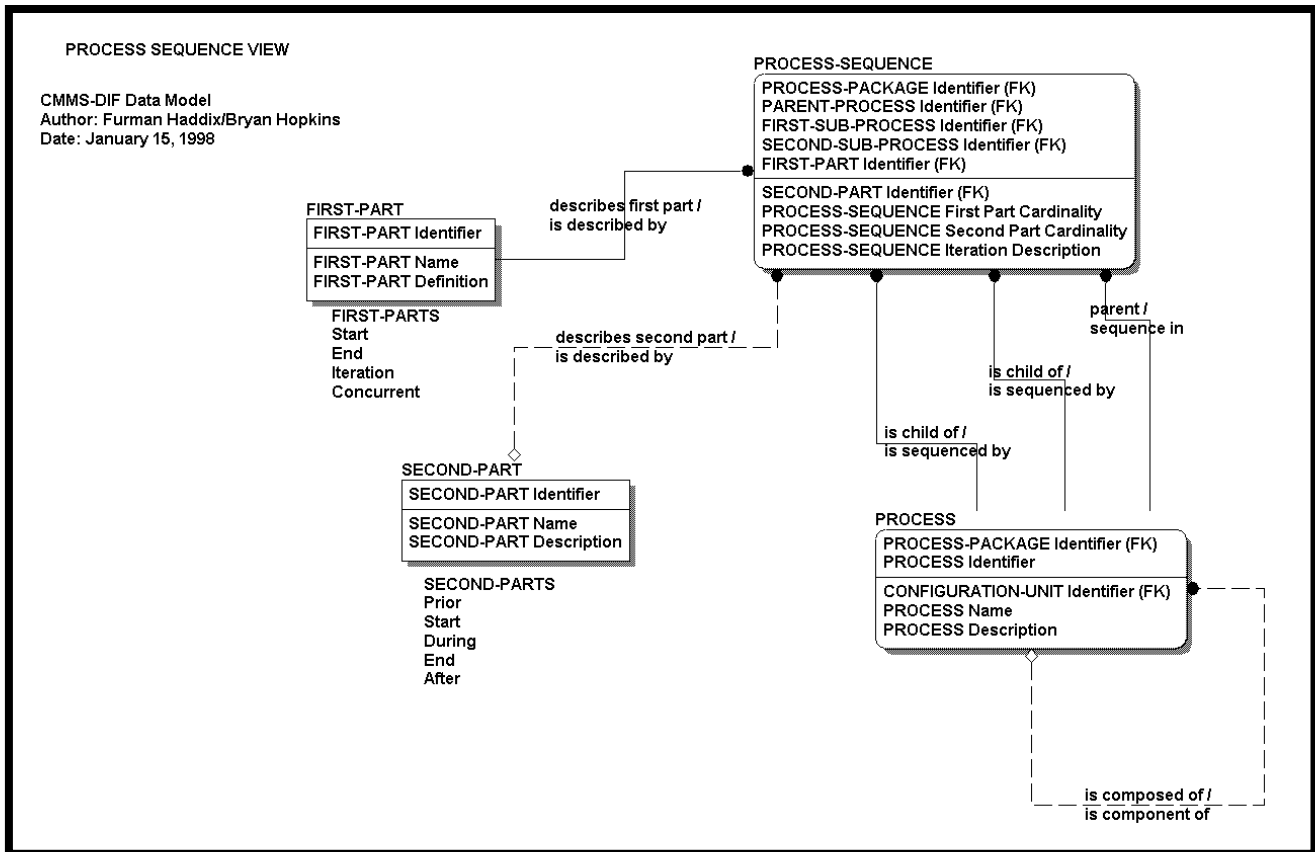
Definition – A Sequence defines a temporal relationship between two Tasks, Actions, or Processes.

The following Task Sequence View illustrates Task Sequence usage, the following Action Sequence View illustrates Action Sequence usage, and the following Process Sequence View illustrates Process Sequence usage.

PRELIMINARY DRAFT COPY



PRELIMINARY DRAFT COPY



5.5 Condition

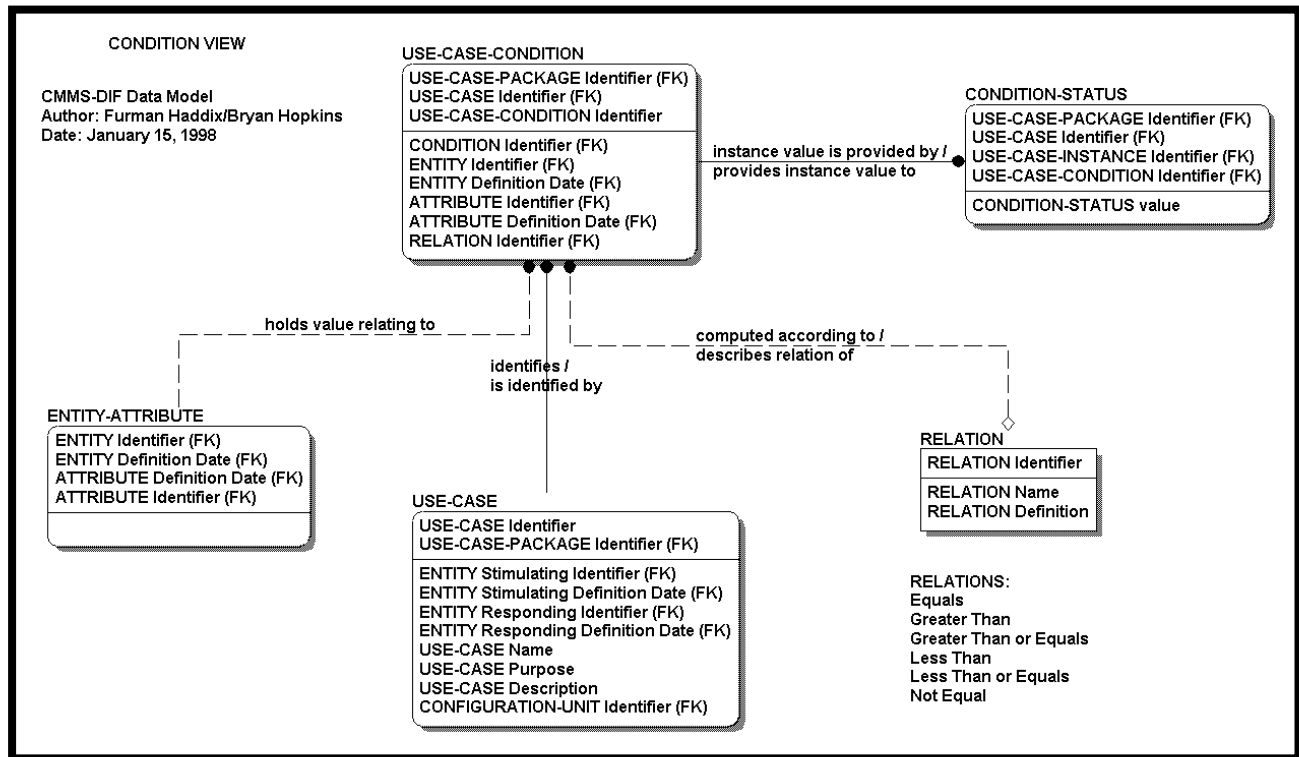
Condition:

Discussion – A condition is a variable in an environment or situation that may affect performance of a task. The three major categories of these conditions are conditions in the physical environment, conditions in the military environment, and conditions in the civil environment. We give a discussion and list of these conditions in Appendix C (“Environment”). Here it is important to note how these conditions may affect task performance. In section 5.3 (“Task”) we enumerate three types of conditionals: IF, WHEN, and WHILE. IF means that if the predicate is true, then the behavior is this, else do another behavior. WHEN means that you will do this; however, you do not do it until the predicate evaluates to true. WHILE means as long as this predicate is true, do (until task completion), else do another activity (until task completion). The conditions (conditions in the physical, military, or civil environment) may act as predicates for these conditionals. Conditions may therefore affect performance of tasks in two major ways. First, they may simply affect how well one is able to perform a task. Second, in acting as predicates for conditionals, they may affect whether or not a task is performed, which task is performed, the time at which a task is performed, or how long a task is performed.

PRELIMINARY DRAFT COPY

Definition – A *Condition* is a variable of an operational environment or situation in which an organization, an equipment, or a person is expected to operate that may affect performance [4].

The following Condition View illustrates Condition usage.



5.6 Use Case Instance, Condition Values, and MOP Standards

Use Case Instance:

Discussion – Use Case Instances are the principal mechanisms DoD uses to construct studies. Use Case Instances provide illustrative environments, starting conditions, and assumptions critical to drive analysis. Use Case Instances put problems to be studied in a plausible political/strategic and military/operational context. In software development, Use Case Instances describe the conditions within which battlespace entities interact with each other. Use Case Instances put software development into an appropriate context (JWARS/DoD) [5]. This concept encompasses a description of a major or lesser regional contingency (MRC or LRC) as a Use Case Instance. A Use Case Instance is effectively a tag for more detailed specifications providing an audit trail of what the specification relates to. A Use Case Instance should effectively define actors, subjects, and conditions. When we refer to conditions here, we are referring to actor and environmental states as of the start of the Use Case Instance. A second purpose of a Use Case Instance is to provide contextual richness to an operation; however, this is not a requirement. If a Use Case Instance is on the

PRELIMINARY DRAFT COPY

scale of a MRC, it will probably have a small number of actors relative to the number of subjects. If the Use Case Instance covers operations of a company command post, it will probably have many actors relative to the number of subjects. If a generic operation is to be used in conjunction with the Use Case Instance, the Use Case Instance must also provide the translation between generic and specific entities. Note with respect to conditions that the Use Case Instance is defining the existing conditions, e.g., it's snowing. We will refer to the use of conditions in terms of flow control as conditionals. Note further that a Use Case Instance can be a configuration unit and that a Use Case Instance from a different configuration unit can be used to provide context for operations. Note that the presence or absence of a Use Case Instance is unrelated to the existence of a domain object model (DOM), as the DOM can be derived from other KLS, if necessary. A Use Case Instance may specify standards, or standards may be determined at the task level, or both.

Use Case Instances also have a high variability in level of resolution, corresponding to that of Use Cases. A Use Case Instance is a description of a context, e.g., Korean peninsula MRC (Major Regional Contingency), which may have a large amount of detail concerning civil, military, and physical conditions, e.g., political unrest, specific orders of battle (friend/foe), massive flooding. A Use Case Instance does not change the Use Case, but it may change the flow of execution. For example, if a bridge is out, engineers must be called in. Once the conditions of Use Case execution are known, standards may be set for Use Case performance and associated with the Use Case Instance. It is also of note that measures of performance (MOP) should be tied to the specific conditions defined by a Use Case Instance. For example, if it takes a certain number of days to build a bridge in good weather, it is likely to take longer to build a bridge in bad weather. Measures of performance must differ with differing conditions. A reasonable measure of performance under a certain set of conditions may not be reasonable under another. Moreover, knowing the conditions that can impair the performance of a task may give information concerning the important aspects of performance to be measured. Conditions and measures of performance are the primary things included in a Use Case Instance.

We define Use Case Instance from two points of view. The first part of the following definition is conceptual, and the second part is constructive or technical.

Definition – (a) A Use Case Instance is an instance of a Use Case possibly characterized by instances of physical, military, and civil conditions, a geographic context, orders of battle, and performance standards. (b) A Use Case Instance provides illustrative environment and starting conditions, measures of performance, and assumptions critical to task flow. Use Case Instances provide a plausible political/strategic and military/operational context and describe the conditions within which battlespace entities interact with each other.

Measure of Performance:

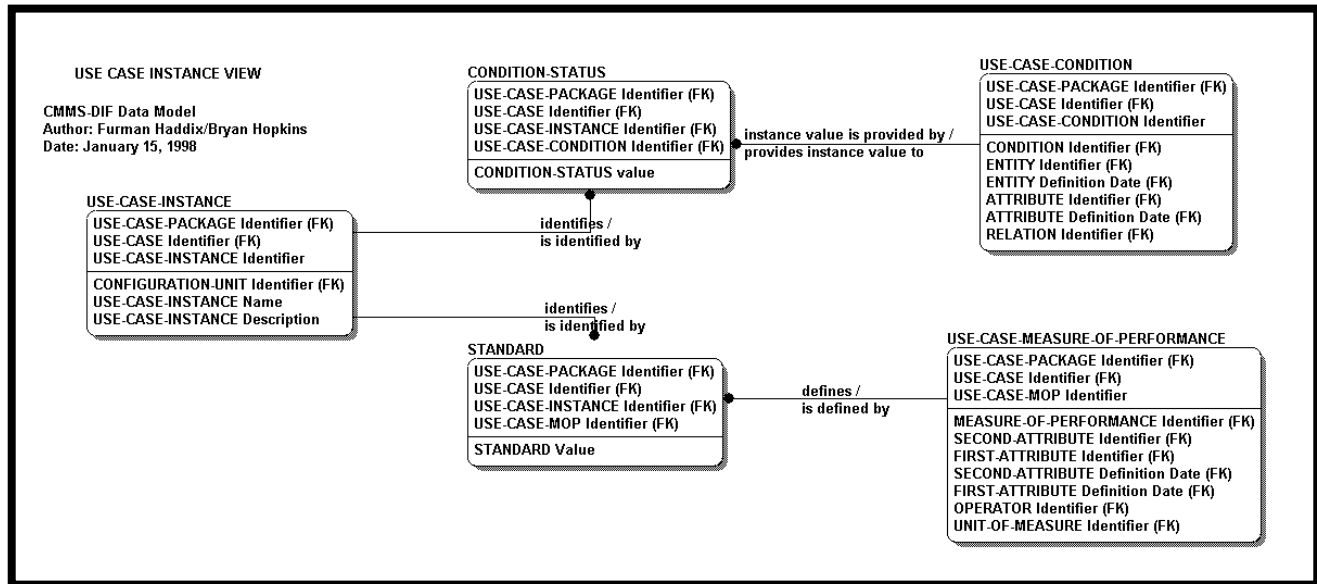
Definition – A Measure of Performance provides a basis for describing varying levels of performance of tasks or missions [4].

Standard:

PRELIMINARY DRAFT COPY

Definition – A standard provides a way of expressing the degree to which an entity must perform a use case under a specified set of conditions.

The following Use Case Instance View illustrates Use Case Instance usage.



5.7 Entity

Entity:

Discussion – An entity is a conceptual object. In general, the entities of CMMS represent real world conceptual objects having significance within the military operations mission space. Entities fall into four categories with regard to their utilization in CMMS.

1. Actor – An actor is a shell. It only provides a stimulus. Normally, no attempt is made to characterize its state or behavior, other than that it is a source of inputs or sink of outputs for the behavior of other entities. With respect to the current modeling effort, it is outside the area of interest, although necessary because of production of an input or consumption of an output. It is needed to rationalize the problem area, or, in other words, to provide a complete view of the activities, but it is beyond the scope of the current area, so it is not specified. One use of actors is in initiating a use case. Also, acting as sources, sinks, and stubs, they limit the scope of use cases by providing black box inputs, outputs, and returns for the articulation of use cases. Although the concept of actor is not directly used in CMMS, it exists therein in the form of incompletely specified entities or stubs. In normal Object-Oriented Programming (OOP) parlance, an actor is a specified element outside of “The System”.
2. Role – A role is the heart of CMMS. Definition of the characteristics and behavior of roles produces the knowledge necessary to support the many user constituencies of CMMS. A role consists of one or more operation specifications and the state

PRELIMINARY DRAFT COPY

variables (attributes) necessary to support them. It is an entity that defines the function provided by, the part played by, or the character assigned to an entity in a process. For example, a Watch Officer in a Joint Operations Center (JOC) is a role that may be played at different times by different officers. In CMMS, this is represented as an entity that represents a type of entity. In the OOP world, a role would be a virtual class.

3. Player – A player is the product of CMMS. A player is an element that a simulation developer might want to implement in a simulation. A player is a composite of all the roles that an agent might perform, for example, J-3, CJTF (Combined Joint Task Force). A role of a J-3 is a JOC Watch Officer. In the OOP world, a player would be an object class, that is, a class that will be instantiated in “The System” implementation.
4. Inactive – An inactive entity is an entity that is part of the use case package but that has no volition of its own and is incapable of taking action based on stimulation. The state of an inactive entity is only changed by manipulation or operation by an active entity, i.e., actor, role, or player.

The following examples may clarify the above distinctions.

1. In many use case instances activity is initiated through the agency of a national official. The President may give an order to the Chairman of the Joint Chiefs of Staff. In this case, the President would be an actor since his activity before and after giving the order is beyond the scope of CMMS. Indeed, in most use case instances the CJCS would also be an actor. Note that an actor in one use case instance may be a role or player in another.
2. In Joint Task Force (JTF) operations, an important set of activities is performed by the Joint Operations Center (JOC) watch officer. This entity should be a role since one would like to see its characteristics and capabilities shared by the three or four officers sharing the responsibility. This is predicated on the fact that the other responsibilities of the three or four officers differ, e.g., only one of the watch officers will be the JOC Director.
3. The Commander of a JTF will be a player in most use case instances. Nevertheless, many of his characteristics and capabilities will be inherited. Examples of contributing roles might include individual, soldier, officer, or battlefield commander. In a particular implementation, some of the above might be considered to be foundation classes.
4. Inactive entities would include entities having no volition. Examples include ammunition, supplies, equipment, reports, features, and facilities. Note that all of these have some state characteristics, albeit limited to location and degree of damage. With regard to capabilities, it is an implementation decision whether bridges change themselves or are changed by external agents.

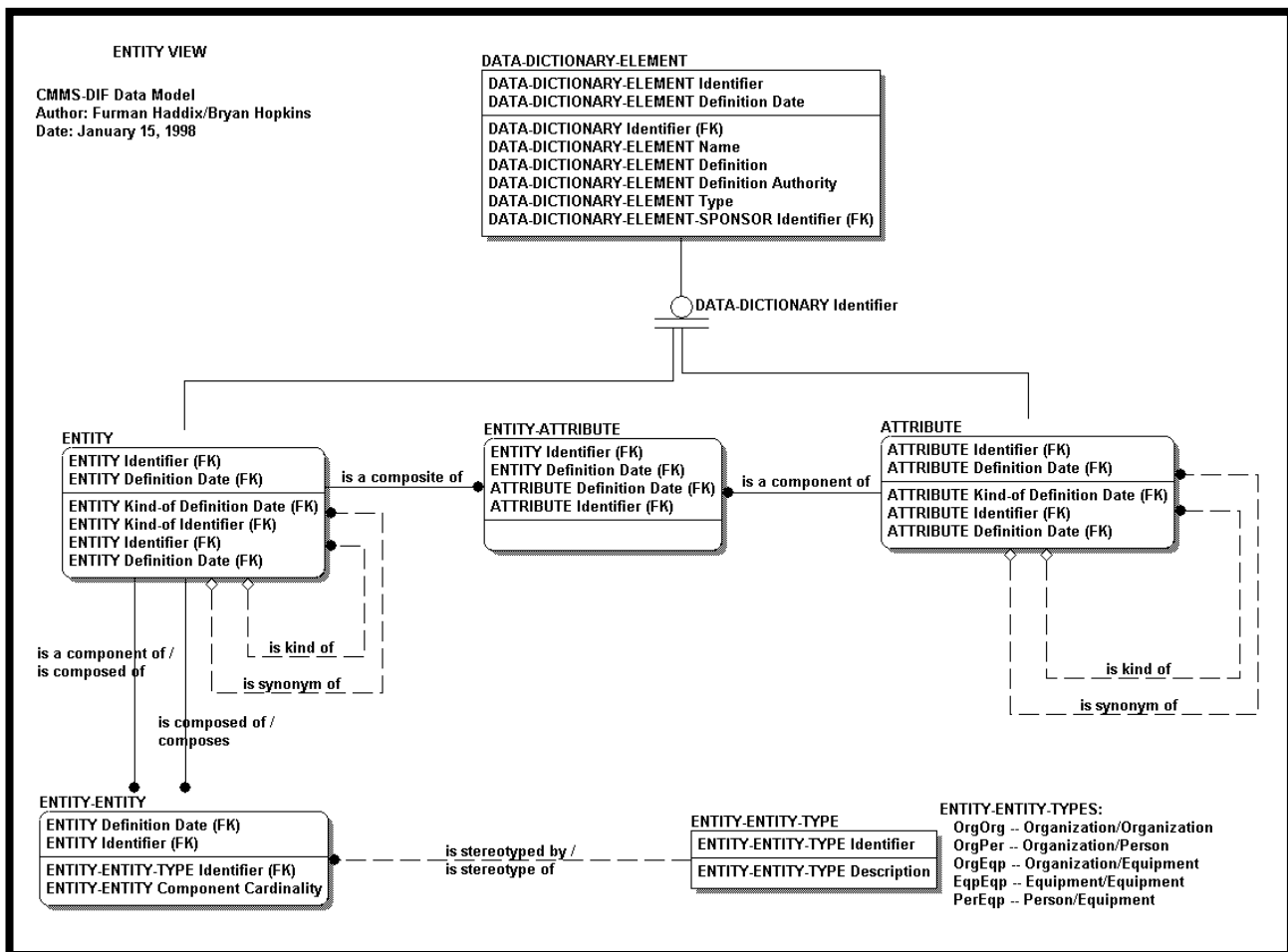
PRELIMINARY DRAFT COPY

- A more difficult issue is how to handle environmental variables, such as weather. Based on the fidelity desired for the simulation, weather areas may be set up; often these are given the status of players by simulation developers, due to the independent pattern of volatility exhibited by weather systems.

A significant aspect of entities is inheritance. Hence, a task may be characterized as being performed by a role entity. That behavior may then be inherited by a player entity. The role entity does not appear in a simulation; the player does.

Definition – An entity is “a distinguishable person, place, thing, or concept about which information is kept. In particular, entity includes the notions of PERSON, ORGANIZATION, FACILITY, FEATURE, MATERIEL,” [6] NETWORK, INFORMATION, and EQUIPMENT.

The following Entity View illustrates Entity usage.



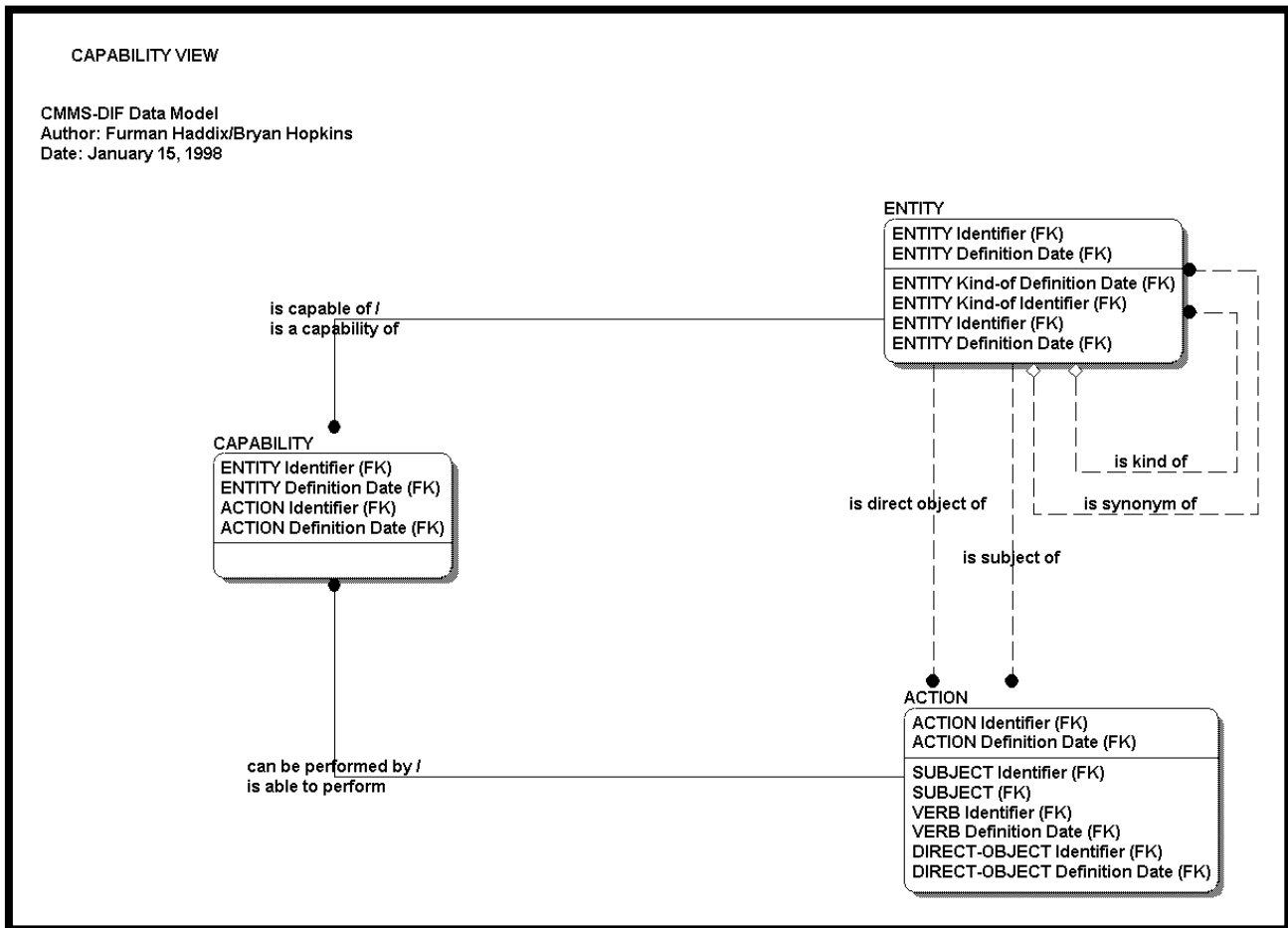
PRELIMINARY DRAFT COPY

Capability:

Discussion – An Entity may have associated with it the capability to perform an Action.

Definition – *Capability is an Entity's ability to perform a certain Action.*

The following Capability View illustrates Capability usage.



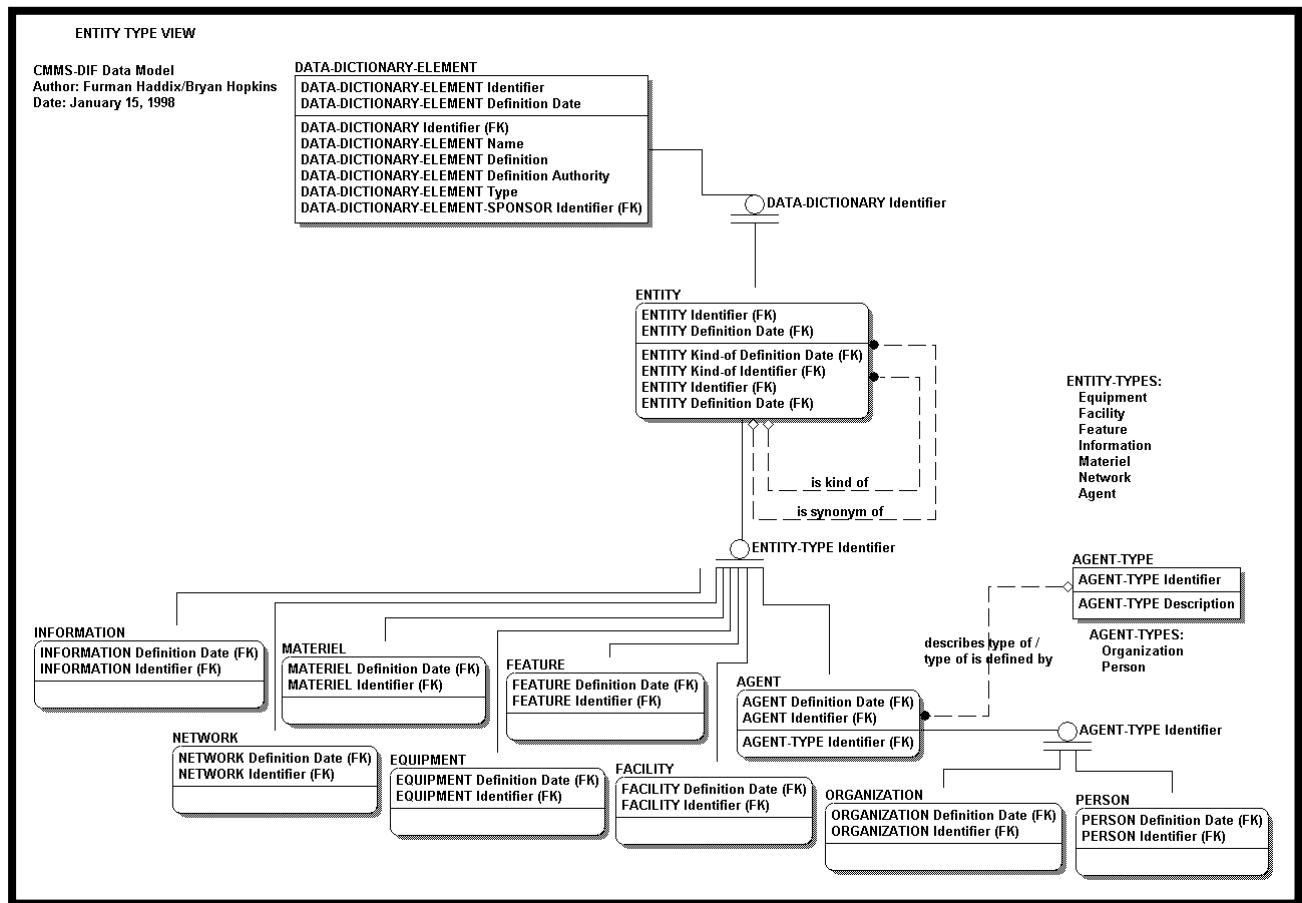
PRELIMINARY DRAFT COPY

Entity Type:

Discussion – Entities may be classified according to function. We use the term Entity Type to refer to a functional type of Entity. These types include Equipment, Facility, Feature, Information, Materiel, Network, Organization, and Person.

Definition – *An Entity Type is a functional type of Entity.*

The following Entity Type View illustrates Entity Type usage.



Two Entity Types of importance are Information and Network.

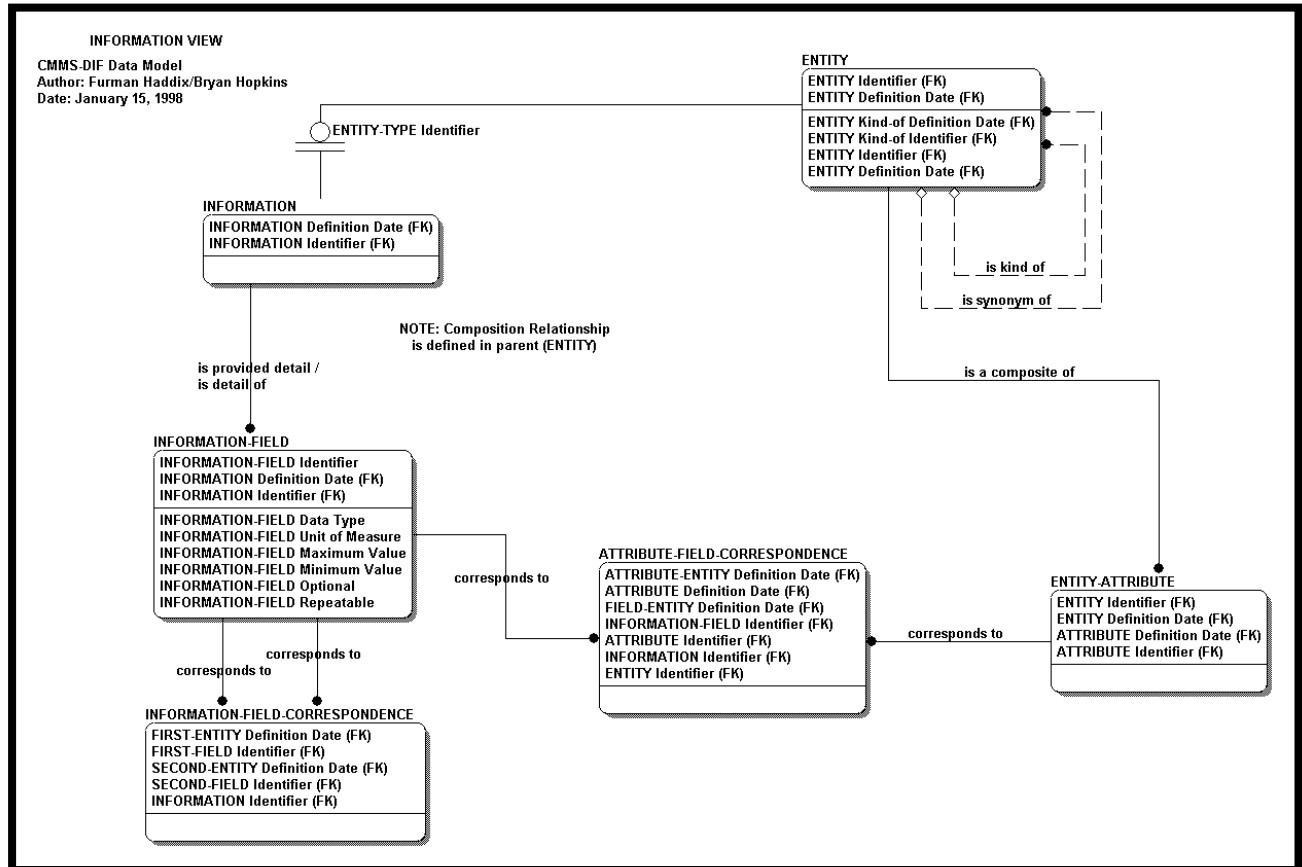
Information:

Discussion – Information refers to knowledge, data, or facts about particular events or situations sent from one Entity to another.

PRELIMINARY DRAFT COPY

Definition – (a) Information is knowledge, data, or facts about particular events or situations sent from one Entity to another. (b) Information is an entity that constitutes an information element or component, as information elements can contain other information elements. The purpose is to provide formatting for messages, reports, plans, orders, etc.

The following Information View illustrates Information usage.



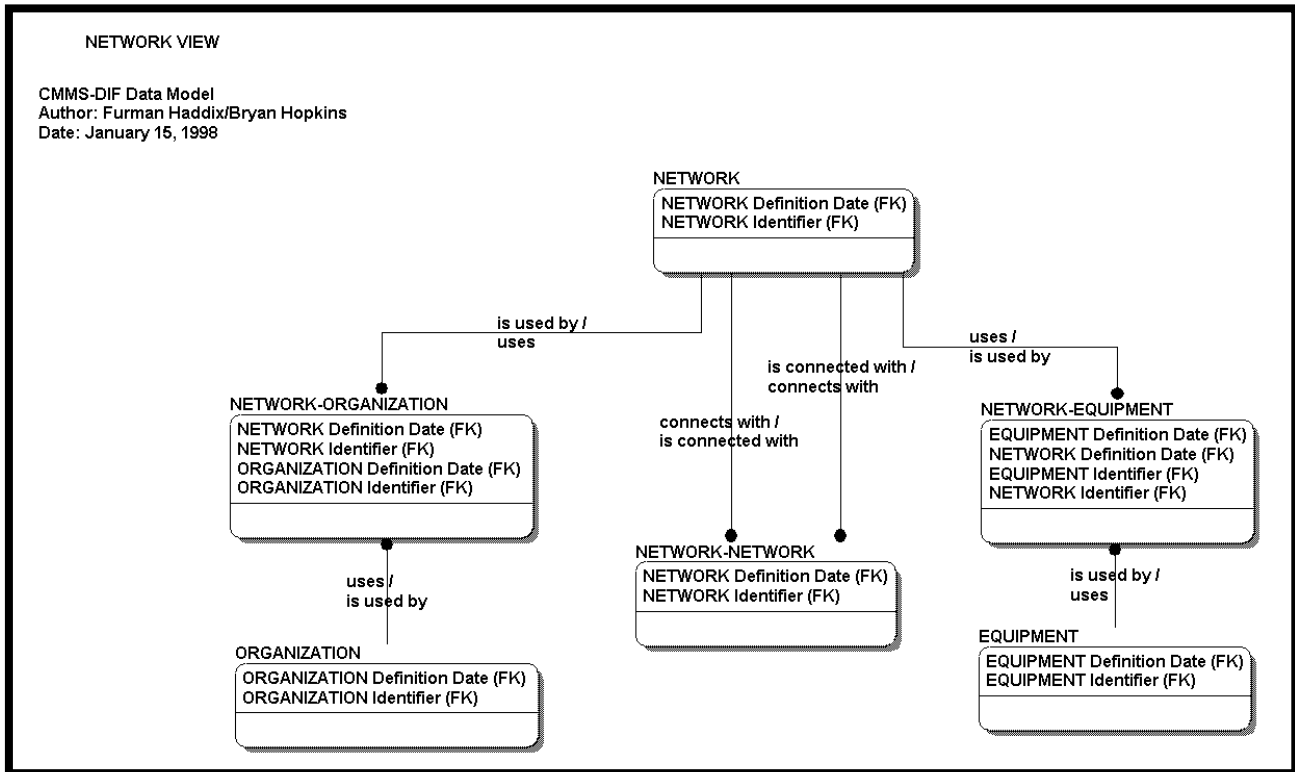
Network:

Discussion – Network in the context of CMMS refers in general to communications networks. A Network may be a computer network but may also encompass cellular networks, radio networks, and voice and digital communications.

Definition – A Network is a system of communications.

The following Network View illustrates Network usage.

PRELIMINARY DRAFT COPY



5.7.1 Rank

Rank refers to traditional military rank, e.g., Colonel, Lieutenant, Ensign, or Civilian. The significance of an individual on the battlefield is the combination of rank and position. The relationship between rank and position is that position represents a capability of a rank. J-3 might be a capability of a Major or a Lieutenant Commander.

5.7.2 Classes

In the object-oriented world, roles, positions, and ranks may all be represented as classes. Selection of which roles, positions, and ranks are categorized as classes is an implementation decision, as is the decision as to which classes are instantiated as objects and which remain virtual classes. Although classes may be the currency of a representation, the classes used in implementation may be quite different, e.g., a representation may differentiate between crew members and tank platform; an implementation may be of a (crewed) tank.

5.7.3 Objects

Objects have two uses in CMMS. They are the implemented instantiation of classes, so they are the objects of object-oriented programming. In knowledge acquisition/engineering, conceptual objects are often used; these are objects of the real world in the non-specialized sense of objects.

PRELIMINARY DRAFT COPY

5.8 Association

Discussion – An association differs from an interaction in that it is of long duration; that is to say that it is reasonably expected to last longer than a task. It also differs in that while interactions conceptually are between tasks, associations conceptually are between entities, as with inheritance.

It was thought that the types of relationships involved in the military operations mission space were sufficiently limited that they could be completely enumerated. The following is a first attempt at a complete enumeration. Additions, deletions and modifications are expected. Some overlap exists among some of the concepts. For example, communication is implicit in command, and occupancy is implicit in occupation. This overlap is desirable since one should not have to use both relationships between two entities to complete the context.

Definition – *An association is a relationship between two entities indicating an area of common interest. Associations include adjacency, collaboration, command, communication, control, occupancy, occupation, operation, opposition, organization, possession, supply, and support.*

Adjacency:

Discussion – Adjacency is a spatial relationship between two entities delineating the possibility of cooperation or attack. It may indicate that an entity is within range of physical sight or hearing of another entity, that it can communicate with another, that it is in firing range of another, that it may be perceived in some way by another, or that it may support another. Whether or not an entity is in support depends on the agent, and whether or not an entity is in range depends on the equipment. Adjacency is not a military relationship, but it is an aid in describing CMMS.

Definition – *Adjacency is the spatial relationship between two entities delineating the possibility of cooperation or attack.*

We may define the following subcategories of adjacency.

a) *In line of sight:*

Definition – *Line of sight is the spatial relationship between two entities in which one entity is within physical sight or hearing of another.*

b) *In line of support:*

Definition – *Line of support is the spatial relationship between two entities in which one entity may support another. This one entity may not actually be in support of the other, but the capability of support exists.*

c) *In range:*

PRELIMINARY DRAFT COPY

Definition – *In range is a spatial relationship between two entities in which one entity may fire upon the other.*

Collaboration:

Discussion – This was a concept that we thought might be useful, particularly in interservice or international operations where the relationship could not be described as command or support and at least nominally was not opposition.

Definition – *Collaboration is the relationship between agents in which there are communication and some commonality of purpose but where the stronger relationships of command or support are not present.*

Command:

Discussion – Command refers to the abstract relationship between battlefield objects in which one can assign tasks, standards, and objectives to another.

Definition – *Command is the relationship between a superior agent and a subordinate agent in which the superior agent may assign tasks, standards, and objectives to the subordinate.*

Communication:

Discussion – Communication is the relationship between two entities in which information is exchanged. Communication may be person to person, person to materiel, or materiel to materiel. It should be noted that communication, like all other associations, is a relationship. It is not the actual transmission of information over that relationship. The actual transmission is an interaction.

Definition – *Communication is the relationship between two entities in which information is exchanged.*

Control:

Discussion – Control is the relationship between two agents in which one agent determines the behavior of the other through frequent interaction.

Definition – *Control is the relationship between two agents in which one agent determines the behavior of the other through frequent interaction.*

Occupancy:

Discussion – Occupancy is the relationship between an agent and an inanimate entity in which the agent may be characterized as having the same location as the entity by virtue of being in, on, or under the inanimate entity. The criteria for whether or not the agent occupies the inanimate entity varies according to the nature of the inanimate entity, e.g., town, vehicle, hill, or shelter.

PRELIMINARY DRAFT COPY

Definition – *Occupancy is the relationship between an agent and an inanimate entity in which the agent may be characterized as having the same location as the entity by virtue of being in, on, or under the inanimate entity.*

Occupation:

Discussion – Occupation is retention or maintenance of possession of a position or an area by force. Occupation is related to occupancy; however, occupancy involves possession and use whereas occupation involves holding and control. Moreover, occupation is often more relevant to a political entity than to a physical entity. In some cases occupation may refer to a physical entity such as a hill or bridge. However, occupancy often involves pacifying a civil population, and in such cases it is more meaningful to speak of occupancy of a political entity than to speak of occupancy of a location. Some overlap exists between the concepts of occupancy and occupation, and in many cases one may think of occupation as occupancy with potential opposition, either internal, external, or both. For example, occupation of a village might have opposition from the villagers inside the village or from opposing forces outside the village.

Definition – *Occupation is the relationship between an agent and a political or inanimate entity in which the agent retains or maintains possession of and exercises authority and effective control over the inanimate entity.*

Operation:

Discussion – Operation is the relationship between an agent and an inanimate entity in which the agent interacts with the entity in order to moderate the entity's behavior. Operation carries with it the idea of making an inanimate entity function. For example, one may operate an aircraft.

Definition – *Operation is the relationship between an agent and an inanimate entity in which the agent interacts with the entity in order to moderate the entity's behavior.*

Opposition:

Discussion – Wars are fought against opponents. This relationship explicitly defines the opponents of interest.

Definition – *Opposition is the relationship in which one agent directly or indirectly attempts to prevent the achievement of the objectives of the other agent and vice versa.*

Organization:

Discussion – Organization is the abstract association of the elements of a command. In that sense it is redundant information with Command. An organization here has the narrow military meaning. Agent is the notional concept of a person or an organization.

Definition – *An organization is the association of the agents of a command, including at least a commander and one other person or organization.*

PRELIMINARY DRAFT COPY

Possession:

Discussion – Possession is the abstract relationship of legal right to limited manipulation and disposal of materiel or facilities.

Definition – *Possession is the relationship between an agent and an inanimate entity that conveys the right of utilization and/or disposition of the entity.*

Supply:

Discussion – Supply involves “the procurement, distribution, maintenance while in storage, and salvage of supplies, including the determination of kind and quality of supplies” [7]. Supply is the relationship between two agents in which one agent provides essentials to the other.

Definition – *Supply is the relationship between two agents in which one agent provides essentials to the other.*

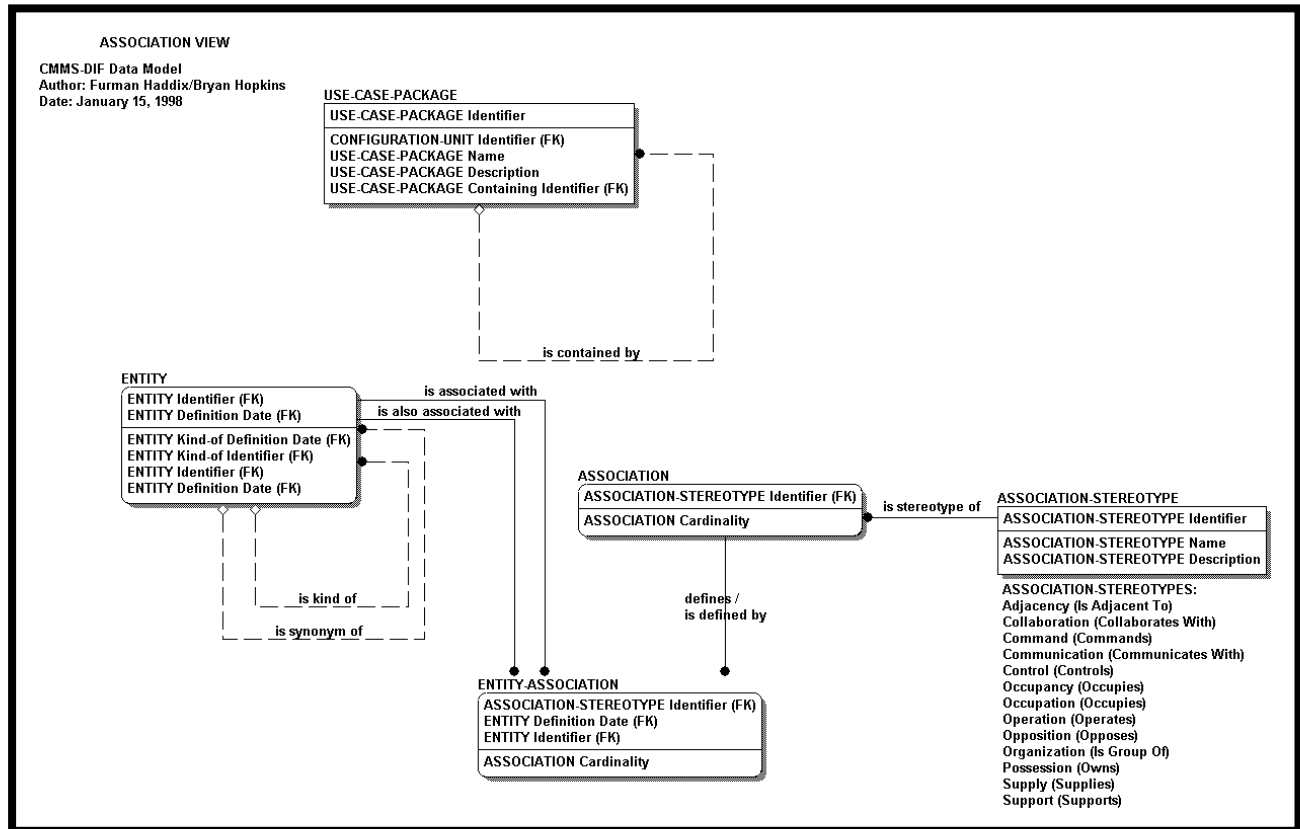
Support:

Discussion – This was intended to convey support both in the sense of a peer agent supporting an attack by another agent and functional support such as indirect fire.

Definition – *Support is the relationship between agents in which one agent’s objective is to facilitate the achievement by the other agent of its objectives.*

The following Association View illustrates Association usage.

PRELIMINARY DRAFT COPY



5.9 Interaction

Interaction:

Discussion – An interaction is the interface that defines the flow of Events, State, Entities, or Tasks between two Entities or Tasks. Although the flows are conceptually between entities, in actuality there must be a sending task and a receiving task. For example, if an entity does not receive a message, it is lost. This requirement can be resolved by using message queues; however, behavior of the entity may be affected by receipt or non-receipt of the message so that timing of receipt is also important.

Many tasks have inputs and outputs. A particular report, for example, may pass through several tasks before reaching its final destination. Interactions are used to specify the path of the report. For example, if we look at a report (A Report) which is produced by one organization and used by five others, we might see the following input and output specifications.

Entity A performs Task A; Entity B performs Task B; ... ; Entity E performs task E.

Task A:

Output: A Report

PRELIMINARY DRAFT COPY

Task B:	Input: A Report	Output: A Report
Task C:	Input: A Report	Output: A Report
Task D:	Input: A Report	
Task E:	Input: A Report	

Figure 4 shows some of the possible ways these inputs and outputs could be arranged. This is why interactions must be made specific. The problem gets even thornier when multiple layers of task decomposition exist. If we assume that the first interpretation is the correct one, then the interactions could be specified as follows:

Task A → A Report → Task B

Task B → A Report → Task C

Task C → A Report → Task D

Task C → A Report → Task E

PRELIMINARY DRAFT COPY

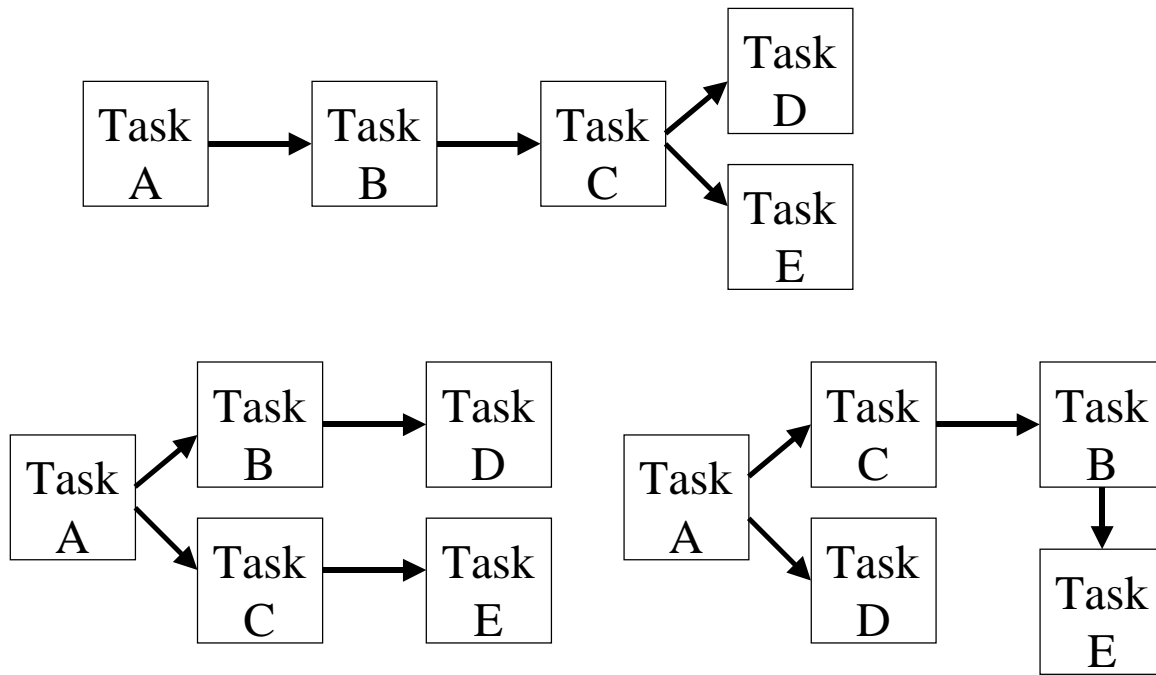


Figure 4. Possible Implicit Interactions

Interactions are characterized by a sender and a receiver. Interactions always may be characterized as a task request, even though a specific request may be only one of several conditions precedent for performance of that task. A common feature of interactions is entity transfer. In general, a transfer of entity or entities is characterized by the transfer type, e.g., asynchronous, sender blocking, or receiver interrupt, and by the entity or entities being transferred. The approach specified here is to treat transfers as being from task to task. In order to avoid a proliferation of transfers to and transfers from tasks, transfers are characterized as part of the “producing” or “consuming” tasks wherever possible. However, the bulk of this characterization and accompanying synchronization is handled through the task sequencing mechanism, so that the interaction need only specify the output (sending) task, the input (receiving) task, and the entity or entities transferred. (UML transfers are the implicit product of a request, an operation specification output, and an operation specification input.) The identification of a task includes the entity performing the task, the task sequence identifier, and the task identifier.

Interactions and associations are essentially the glue that holds the components of CMMS together. The distinction that we make between interactions and associations is that although interactions represent a process or transaction that might take some duration of time, in the abstraction of CMMS they may be thought of as instantaneous, occurring as output at the end of one task and as input at the beginning of another task. Furthermore,

PRELIMINARY DRAFT COPY

with respect again to both interactions and associations, it was felt that the general capabilities provided by most CASE tools were not necessary for CMMS and might, in some cases, be dysfunctional by encouraging multiple representations of a single semantic element.

The following represents an attempt to enumerate the types of interactions. This list claims to be no more than a set of examples, and additional interactions will probably be suggested and included. Note that even though entity actors can be source or target for interactions, rarely would both source and target be entity actors.

It should be noted that interactions are sent over associations in CMMS as in UML. An interaction presupposes an association. The following discussions specify at least one association over which each interaction is generally sent.

Definition – An interaction is the sending of information or other elements between two tasks or agents. The types of interaction include assignment, attack, collision, delegation, land, launch, resupply, transfer, and transmission.

Assignment:

Discussion – The first instance of assignment in an exercise will be when an actor assigns a use case to a subject. Other assignments during a use case will generally be assignments of use cases or tasks to entities. Conceptually, assignment may be thought of as being from task to task, with the receiving task(s) being the first task(s) of the use case, and in the case of assignment by an actor, the sender is an implicit task since the actor has no explicit tasks. Assignment takes place over the association “Command”.

Definition – An assignment is the representation of an initiation of a use case thread for a subject as an output of a task or action by an authority entity. An attribute of an assignment is a use case (thread).

Attack:

Discussion – An attack can take place when an agent and a target are in opposition and when the agent is in range of the target. This is the time when casualties and/or damage are assessed. Attack takes place over the association “In range”.

Definition – Attack is the interaction in which projectiles or other potentially damaging materiel are sent from an agent to another entity.

Collision:

Discussion – A collision occurs when two or more objects try to occupy the same space at once. The name for this interaction may be used to refer either to a collision of a vehicle with a vehicle, to a collision of a projectile with a vehicle, to a collision of a projectile with a person or persons, or to a collision of a projectile with other entities such as a bridge or building. Collision takes place over the association “Adjacency”.

PRELIMINARY DRAFT COPY

Definition – *Collision is the interaction in which two or more objects attempt to occupy the same space at once.*

Delegation:

Discussion – A delegation is the assignment of command authority over an agent from one agent to another. Thus, delegation has an attribute of the Entity for which command was delegated. Delegation takes place over the association “Command”.

Definition – *Delegation is the interaction in which command and control of a subordinate agent are assigned from one superior agent to another.*

Land:

Discussion – To land is to cause an aircraft to return to surface mode from flight in a controlled and procedurally executed manner. To land may initiate on board status or cause a platform to return to land. It is important to note that if an aircraft lands on a carrier, it becomes part of that carrier. Land takes place over the association “Operation”.

Definition – *Land is the interaction that causes an aircraft to return to surface mode from flight in a controlled and procedurally executed manner.*

Launch:

Discussion – To launch is to transition an entity from surface mode to dynamic flight. To launch may terminate on board status. The term may be used of aircraft, spacecraft, or projectiles. Launch takes place over the association “Operation”.

Definition – *Launch is the interaction that transitions an entity from surface mode to dynamic flight.*

Resupply:

Discussion – Resupply is the act of replenishing stocks in order to maintain required levels of supply [7]. Resupply may be automatic, emergency, or planned. Automatic resupply is a resupply mission fully planned before insertion of a special operations team into the operations area that occurs at a prearranged time and location, unless changed by the operating team after insertion [7]. Emergency resupply is a resupply mission that occurs based on a predetermined set of circumstances and time interval should radio contact not be established or, once established, be lost between a special operations tactical element and its base [7]. Planned resupply is shipment of supplies in a regular flow as envisaged by existing preplanned schedules and organizations, which will usually include some form of planned procurement [7]. Resupply takes place over the association “Supply”.

Definition – *Resupply is the interaction that provides additional expendable materiel.*

PRELIMINARY DRAFT COPY

Transfer:

Discussion – This is the interaction reserved for shift of control over materiel or a facility from one agent to another. In either case, it is assumed that there is significant adjacency between the agents. Transfer takes place over the association “Adjacency”.

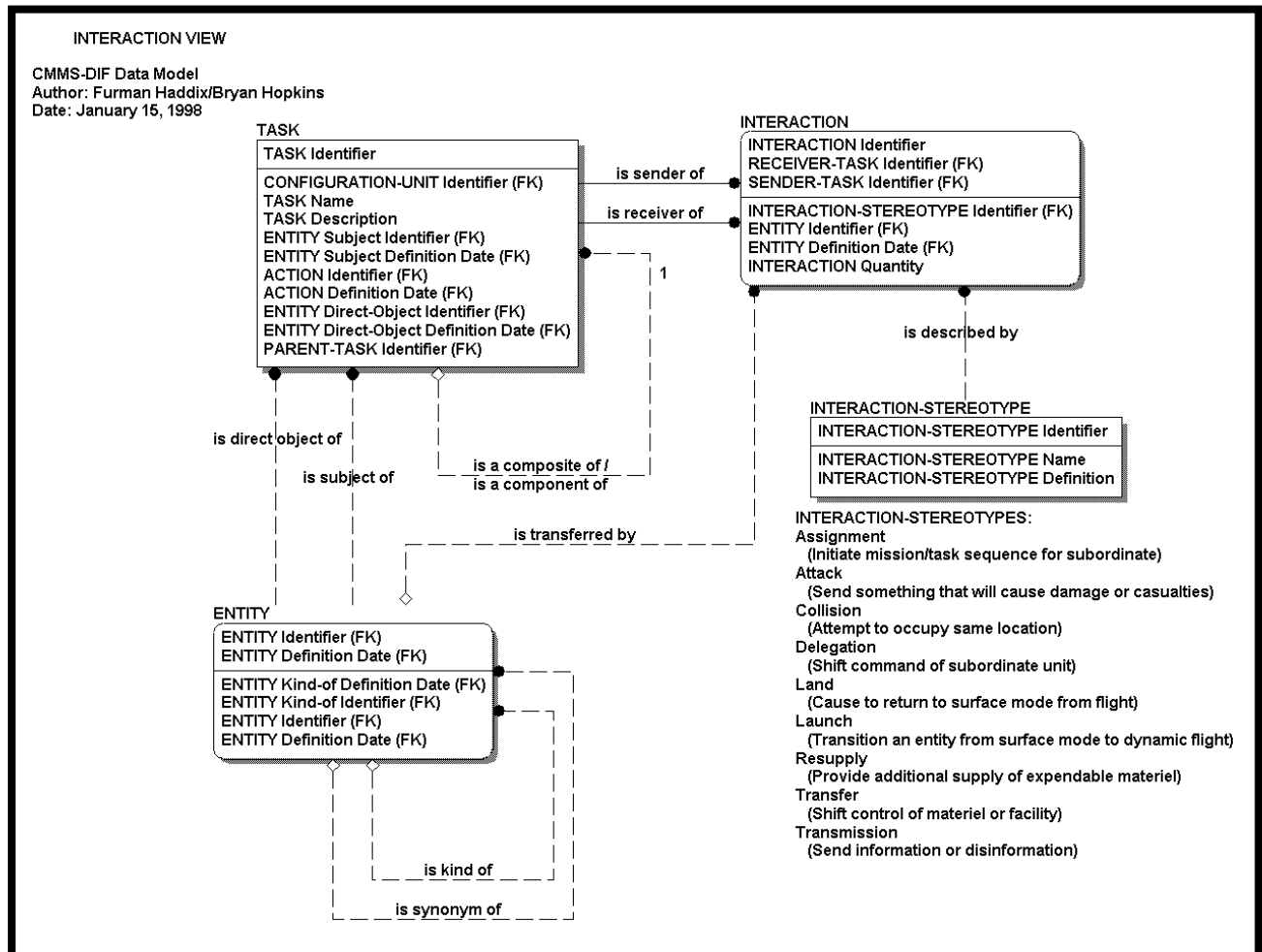
Definition – *A transfer is the interaction used for shift of control over materiel or a facility from one agent to another.*

Transmission:

Discussion – It was decided to use interactions which were specific to one characterization. Whereas several interactions can be characterized as send (and receive), this one was defined to include only information. Transmission takes place over the association “Communication”.

Definition – Transmission is the interaction in which information or disinformation is sent from one agent to another.

The following Interaction View illustrates Interaction usage.



PRELIMINARY DRAFT COPY

5.10 Verb

Verb:

Discussion – Verb is used here as in normal colloquial usage, with a few small differences. The verbs of interest are the verbs used to describe actions of the military operations mission space. A verb is an atomic concept of CMMS. Common usage of verbs is an essential requirement for CMMS. The definition that will be used for verbs in CMMS is limited to transitive verbs. Intransitive verbs are represented by relationships. Verbs are maintained in a Verb Data Dictionary and are constrained in their usage as specified by the Verb Data Dictionary. Multiple meanings may exist for a single verb. Multiple verbs may share the same description. This area of ambiguity is cleared up by the concept of action.

Definition – *A verb is a descriptor of an action performed by an entity in the military operations mission space. For the action to be of interest to CMMS, it must result in a change of state in the entity performing the action or in the entity that is the object of the action.*

5.11 Inheritance

Discussion – Inheritance is included here because it is a common association. CMMS has a limited inheritance; however, it is explicitly rendered in the data structures of the KLs potentially inheriting. Thus, association as it is used here does not include inheritance.

Definition – *Inheritance is the transfer of attributes and behavior from a superclass to a subclass.*

5.12 Aggregation

Discussion – Aggregation is the relationship between components and their assembly. It describes only physical “part of” relationships such as component assembly. An aggregation in CMMS might be an aggregation of a radar system into a plane or of a turret into a tank. Aggregation does not apply to command association. A brigade commander commands battalions, but the battalions and the brigade headquarters are not aggregated into some combination. Organizational components form associations but not aggregations.

Definition – *Aggregation is the relationship between components and their assembly.*

5.13 Configuration Unit

Configuration Unit:

Discussion – The concept of Configuration Unit is very important for several reasons. A configuration unit represents an atomic unit from the aspects of ownership, version control, and configuration management. Configuration units may serve as building blocks when constructing a new use case package. Generally, a configuration

PRELIMINARY DRAFT COPY

unit represents a somewhat homogenous aggregation of knowledge. A configuration unit can vary in size by several orders of magnitude. Key concepts here are that a configuration unit represents a single production effort; it represents a single examination effort; and it represents a single configuration effort in terms of registration, version, and release. This latter indicates that it would necessarily be homogenous with respect to classification. One of the functions of CMMS is to identify production, examination, registration, version, and release characteristics with respect to each conceptual model; the concept of a configuration unit means that this information can be kept for a configuration unit which may contain few or many conceptual object models and relationships between conceptual object models.

Definition – A configuration unit represents a set of related objects and object relationships of arbitrary extent and common production, examination, registration, version, and release characteristics.

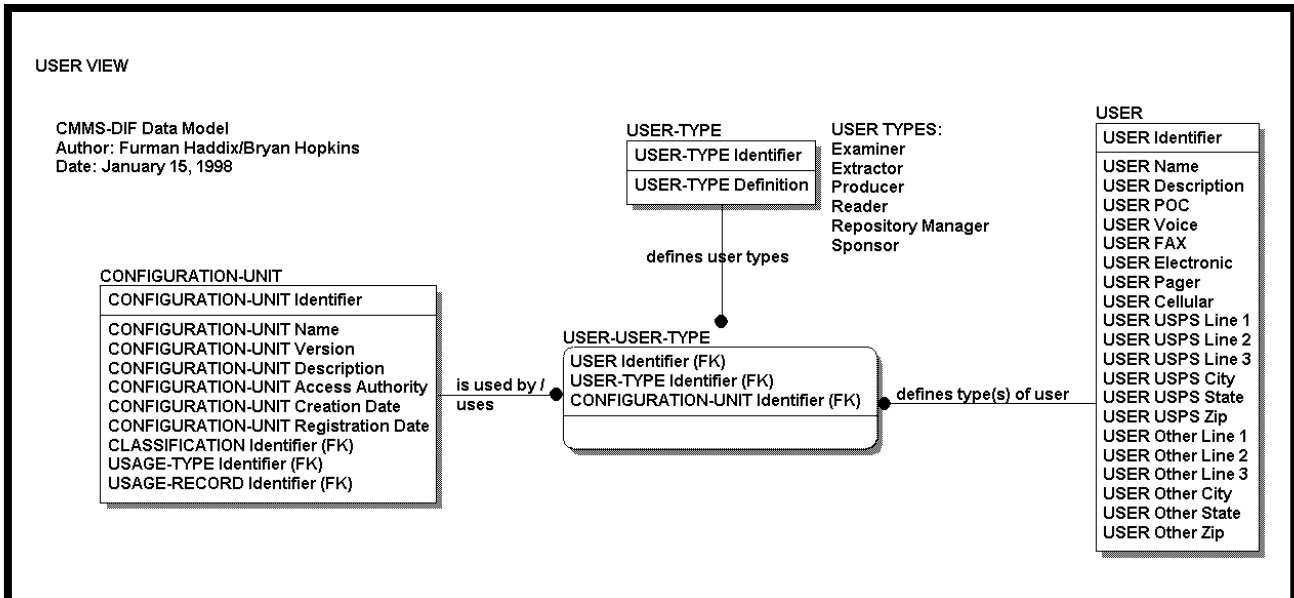
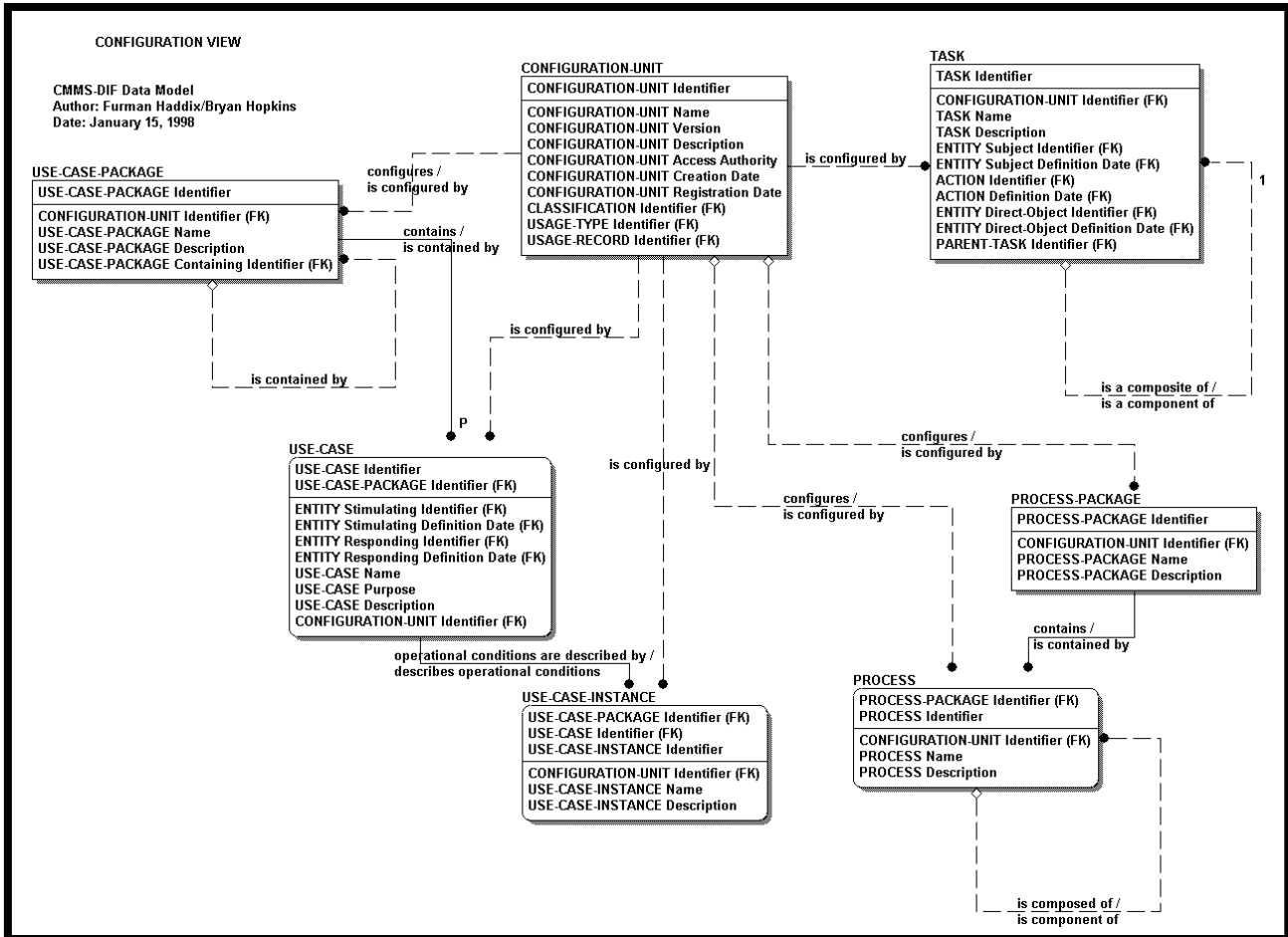
User:

Discussion – A User may use a Configuration Unit as an Examiner, an Extractor, a Producer, a Reader, a Repository Manager, or a Sponsor.

Definition – A User is a person or organization that uses a system, whether as Examiner, Extractor, Producer, Reader, Repository Manager, or Sponsor.

The following Configuration Unit View illustrates Configuration Unit usage, and the following User View illustrates User usage.

PRELIMINARY DRAFT COPY



PRELIMINARY DRAFT COPY

5.14 Data Dictionary

Data Dictionary:

Discussion – The purpose of data dictionary usage in CMMS is to provide a robust foundation for semantics. When a name is used in CMMS, if it is a data dictionary element, assurance should be provided that it will conform to a definition provided in the appropriate data dictionary. Within CMMS, elements in a data dictionary may have one or more definitions; however, the element defined in a data dictionary is considered equivalent to all of its instances, regardless of usage, if its name is the same as the entry in the data dictionary. For example, “provide” is considered to be the same verb, regardless of which definition is applicable. For actions, however, it is unacceptable to have more than one interpretation. All elements of CMMS should be under configuration management and responsible semantics management. The members of data dictionaries and enumerations provide the atomic elements for construction of CMMS, while the data dictionaries and enumeration data dictionaries provide the basis for configuration management and responsibility-based management of semantics of these atomic elements.

Definition – *A data dictionary provides a unique set of semantics for each uniquely identified data element within a context.*

Note – CMMS is considered to be one context with respect to uniqueness of semantics for elements of data dictionaries. It is contemplated that CMMS will have several data dictionaries, including Verbs, Actions, Entities, and Attributes. Additional data dictionaries may be defined to accommodate the need for globally unique nomenclature for other CMMS knowledge elements. Conversely, configuration unit knowledge elements have a name space scope no larger than the configuration unit. The responsibility for consistency and usability with respect to these items lies with the configuration unit sponsor. This approach contemplates that all knowledge elements would belong to either a configuration unit, a data dictionary, or an enumeration data dictionary.

Attributes of CMMS entities are considered to be pure state variables. In other words, if the variable is not modified by tasks, then it should not be specified. Thus, much characteristics and performance (C&P) data would not be included in a CMMS model. CMMS would consider current location, current speed, and remaining range (based on fuel available) to be state variables. The methodology and data to determine speed, location, remaining fuel, etc. are not appropriate concerns of CMMS at this time.

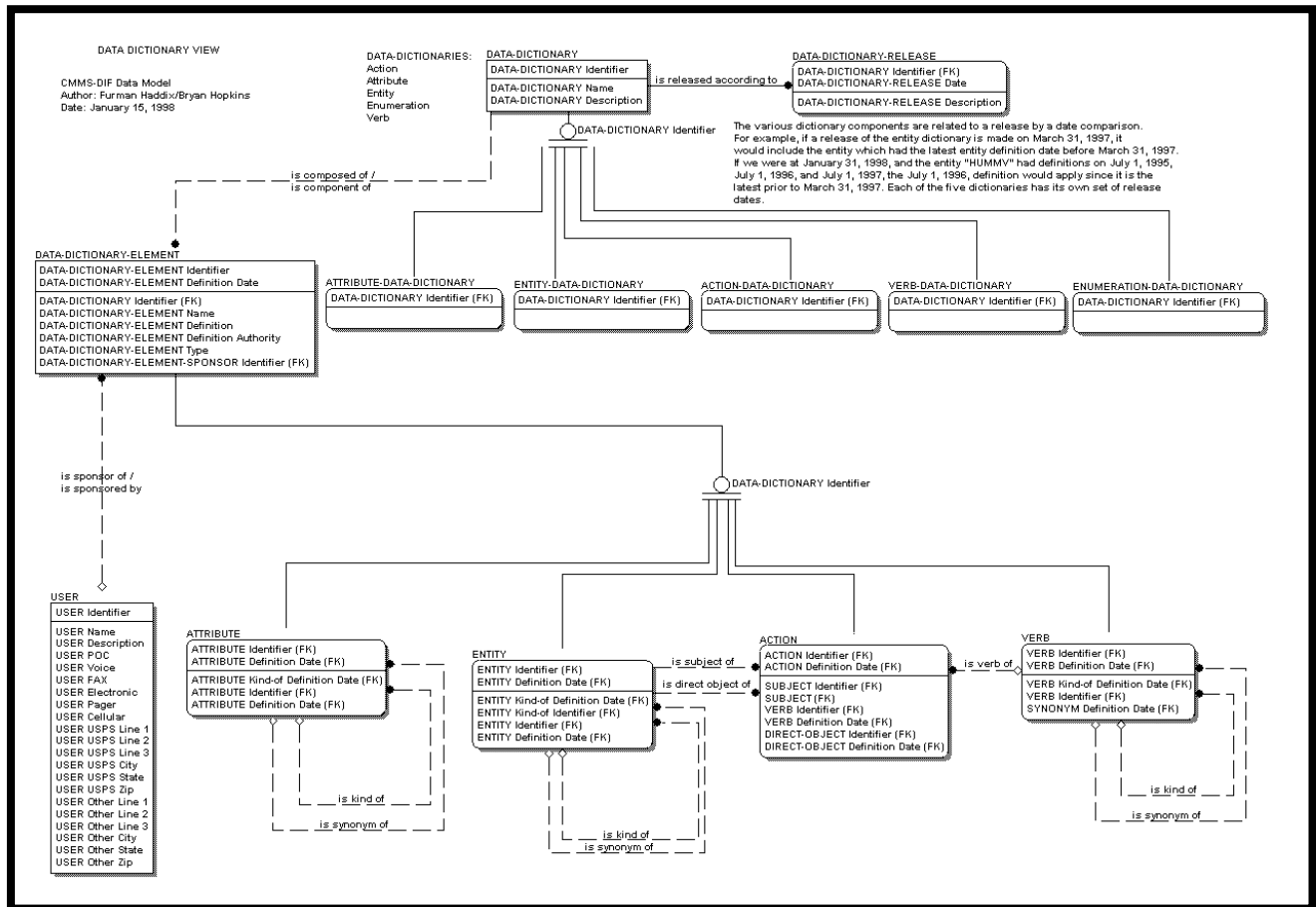
The set of state variables applicable to a specific entity is considered to be part of the definition of that entity and so is under the control of the entity authority. A change in the attributes applicable to an entity is considered to be a change in the definition of the entity.

A possible source of conflict occurs if attributes are inherited based on type. This would seem to be a good feature in that it potentially could greatly reduce the effort

PRELIMINARY DRAFT COPY

required to specify all of the state variables for all entities. There really is not conflict because the “owner” of an entity can decide if that entity is “a type of.” Thus, if the owner of “Abrams Tank” believes that altitude is not a valid attribute, he can terminate attribute type “tank”.

The following Data Dictionary View illustrates Data Dictionary usage.



5.15 Metadata

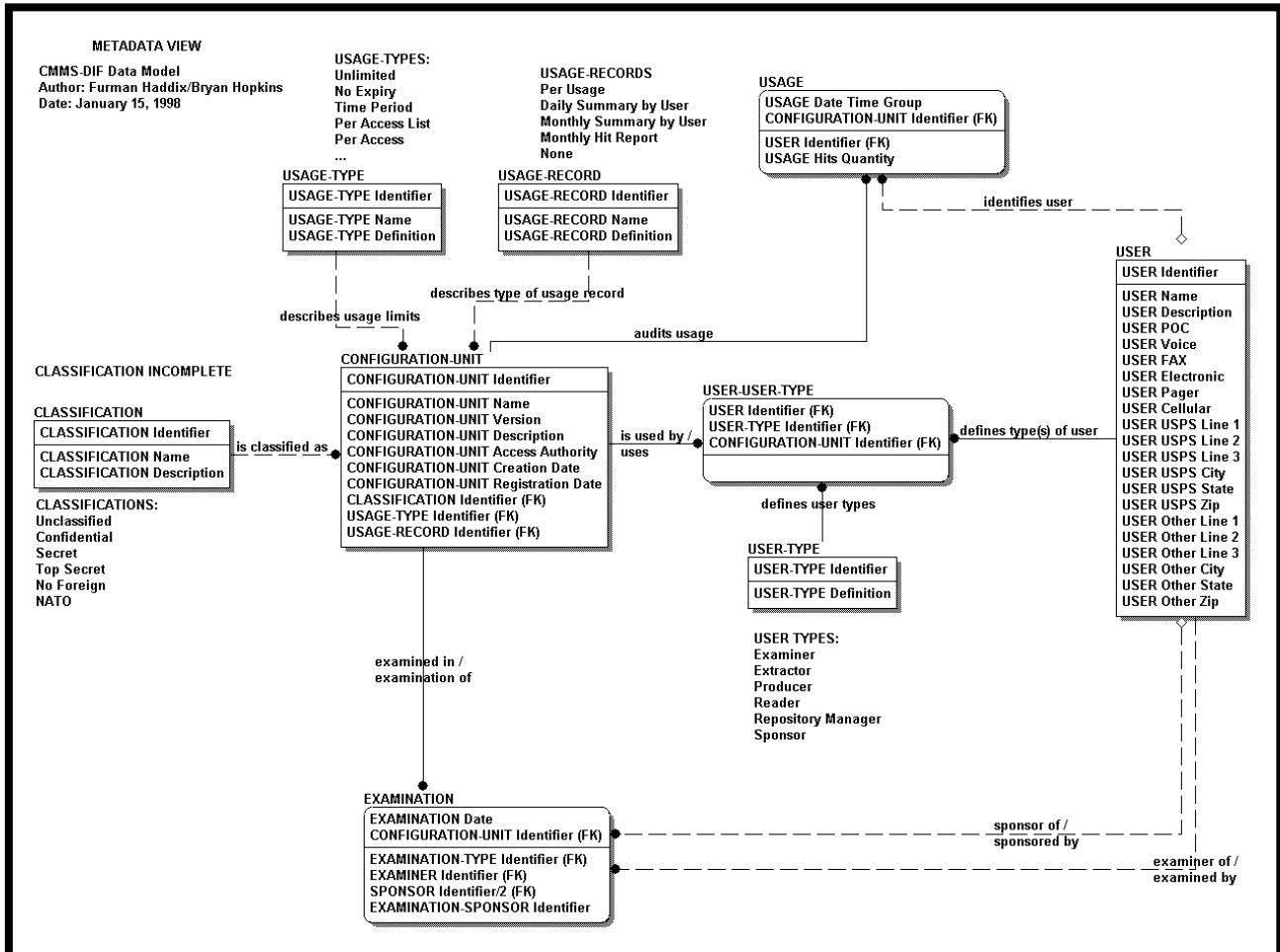
Metadata:

Discussion – Metadata is data about data. Any information about a piece of data except for the data instance value is considered metadata. In general, format of data is that data’s most basic metadata. However, other metadata of interest may include (but are by no means limited to) producer, ownership, production data, and verification and validation procedures applied.

PRELIMINARY DRAFT COPY

Definition – *Metadata is any data about data other than the data instance value.*

The following Metadata View illustrates Metadata usage.



5.16 Enumeration

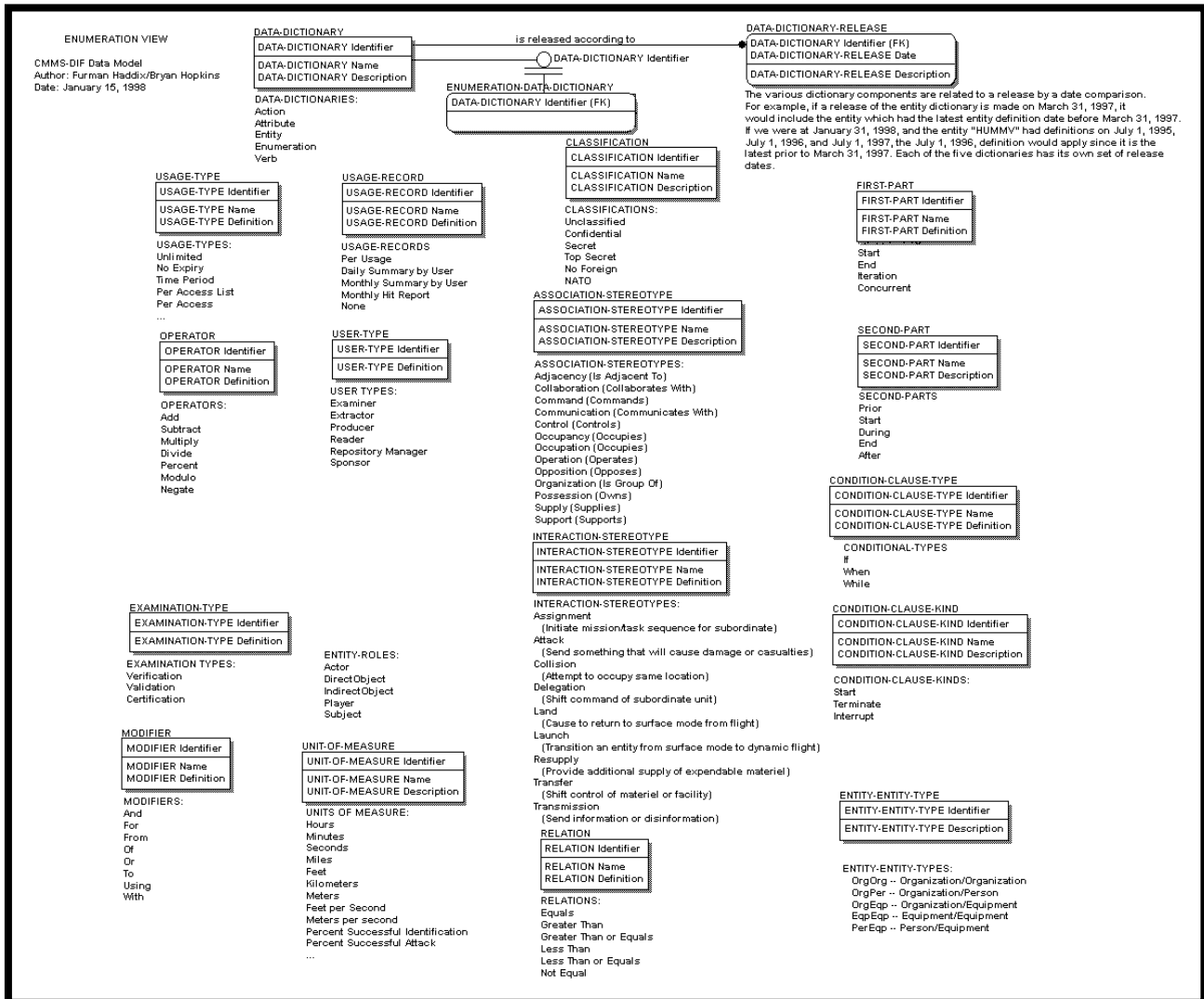
Enumeration:

Discussion – An enumeration of an element gives the set of values that parameters may take on. For example, when we give the enumeration of the element “conditional type” as “if, when, and while,” we are defining the values that an instance of conditional type may have. When we give the enumeration of “classification” as “unclassified, confidential, secret, top secret, no foreign, and NATO,” we are defining the values that a classification may have.

PRELIMINARY DRAFT COPY

Definition – An enumeration of an element gives the set of values that parameters may take on.

The following Enumeration View illustrates Enumeration usage.



5.17 Examination

Examination:

Discussion – Examination consists of three concepts: verification, validation, and certification. Verification is comparison of a model against a specification, a document that describes the specific elements that need to be in the model. In other words, verification is the element of the examination that asks whether or not the model does the job correctly. Validation, on the other hand, asks whether or not the model does the

PRELIMINARY DRAFT COPY

right job. Validation is comparison with a benchmark. The benchmark is a standard of purpose. Validation tests whether or not the behavior exhibited is sufficient to satisfy the purpose of the object being validated. It tests for whether or not the system satisfies its purpose and whether or not the system models the real world from the point of view of the user. Certification in this context means roughly what it means in natural language; certification is the approval of a component of CMMS or a set of related components of CMMS for a specific use. The certification is generally not provided until that component or set of related components has been used.

Examination:

Definition – An examination is a test to determine whether or not a component of CMMS or a set of related components of CMMS is acceptable for a specific use.

Verification:

Definition – Verification is the process of determining that a model or simulation implementation accurately represents the developer's conceptual description and specifications [7].

Validation:

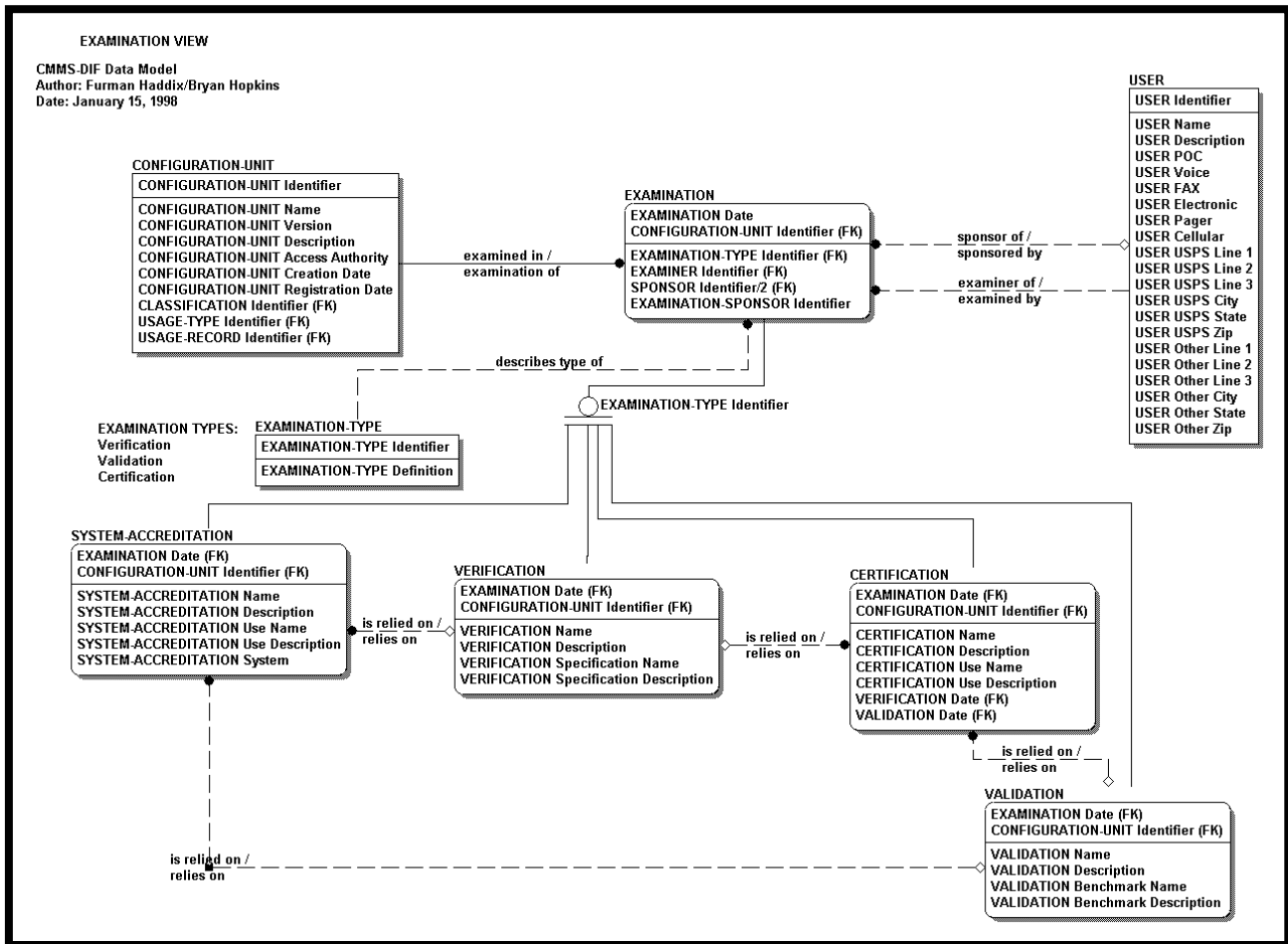
Definition – Validation is the process of determining the degree to which a model or simulation is an accurate representation of the real world from the perspective of the intended users of the model or simulation [7].

Certification:

Definition – Certification is the approval of a component of CMMS or a set of related components of CMMS for a specific use.

The following Examination View illustrates Examination usage.

PRELIMINARY DRAFT COPY



5.18 Reference

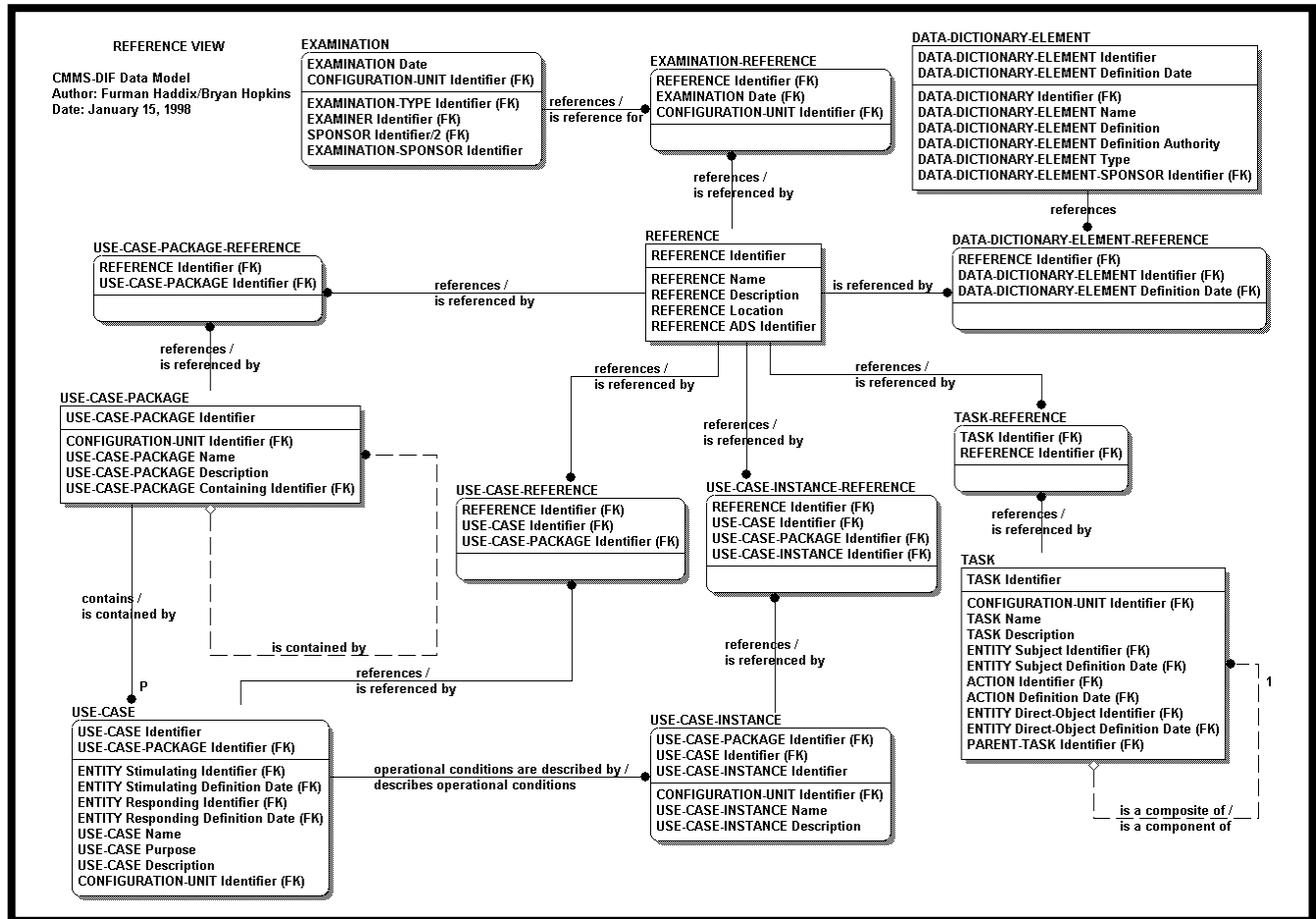
Reference:

Discussion – That references be available for entities is a crucial aspect of the traceability of CMMS as a model. Information found in the model should be traceable to its authority. If one finds information, one should be able to find what authority applied to it. Reference gives the authoritative sources upon which the model is based.

Definition – *A reference provides the authoritative source or sources upon which an element in the model is based.*

The following Reference View illustrates Reference usage.

PRELIMINARY DRAFT COPY



5.19 Conceptual Object Model

System:

Discussion – This is a real world system, as opposed to a computer system. In the military context, this could be an infantry brigade, an aircraft carrier, a multiple launch rocket system, a member of a ground crew, etc. The significance from a CMMS point of view is that the system by implication draws a border between what is of interest within a specific effort and what is outside the scope. (The border is probably broad enough to encompass a sizable “gray” area of items that are not elements of the system but that are significant in defining context for the system.) Much that surrounds the “object” real-world system will have to be provided by the computer simulation system being developed. Because of the overloading of the term “system”, we have substituted conceptual object.

Definition – A system is a collection of connected units that are organized to accomplish a specific purpose. A system can be described by one or more models, possibly from different viewpoints [8].

PRELIMINARY DRAFT COPY

Conceptual Object:

Discussion – Conceptual objects may be mountains, cities, armor battalions, aircraft carriers, pilots, or situation reports. For CMMS, we are more interested in armor battalions, aircraft carriers, and pilots because they may be subjects of tasks. (The aircraft carrier is more significant as an organization of naval personnel than as a platform from the CMMS perspective.)

Definition – *A conceptual object is a first-order abstraction of a real-world object. It may be a collection of abstractly or concretely connected agents that are organized to accomplish a specific purpose. A conceptual object can be described by one or more models or views, possibly from different viewpoints.*

Model:

Definition – *A model is a semantically closed abstraction of a system [9].*

Conceptual Object Model:

Discussion – A simulation developer has been tasked to simulate one or more conceptual objects. Within his tasking are defined the aspects of the conceptual object that are of significance to his simulation. Within CMMS there is a possible scope of the conceptual object model (COM); the actual model available may be null or partial. If it is partial, it may cover all of the aspects of significance to the developer, which is hoped eventually to be a high-probability event, or it may cover some, or none.

Before going any further, we will address the issue, “If we had a conceptual model, how would we control its configuration?” The answer is that a conceptual object model must be part of a configuration unit.

Definition – *A conceptual object model is a semantically closed abstraction of a conceptual object.*

5.20 Attributes

Attributes:

Discussion – An attribute represents information that one needs to know about an entity. Here attribute is used to describe characteristics of real world entities, specifically those characteristics that perform the basis for military decisions. Thus, even though it could be argued that a real world characteristic of a tank is its maximum horsepower, the decision impacting characteristics generally will be attributes such as speed and range.

The attributes of an entity are data that the entity carries concerning the entity. Attributes may be inherited from parent entities. It is important to recognize that attributes may be simple or complex. A simple attribute is atomic; it may be an integer, floating point

PRELIMINARY DRAFT COPY

number, Boolean number, or character. A complex attribute may be a data structure, class, or array. A complex attribute might also be recursively decomposable.

Definition – *An attribute is information about an entity carried by that entity.*

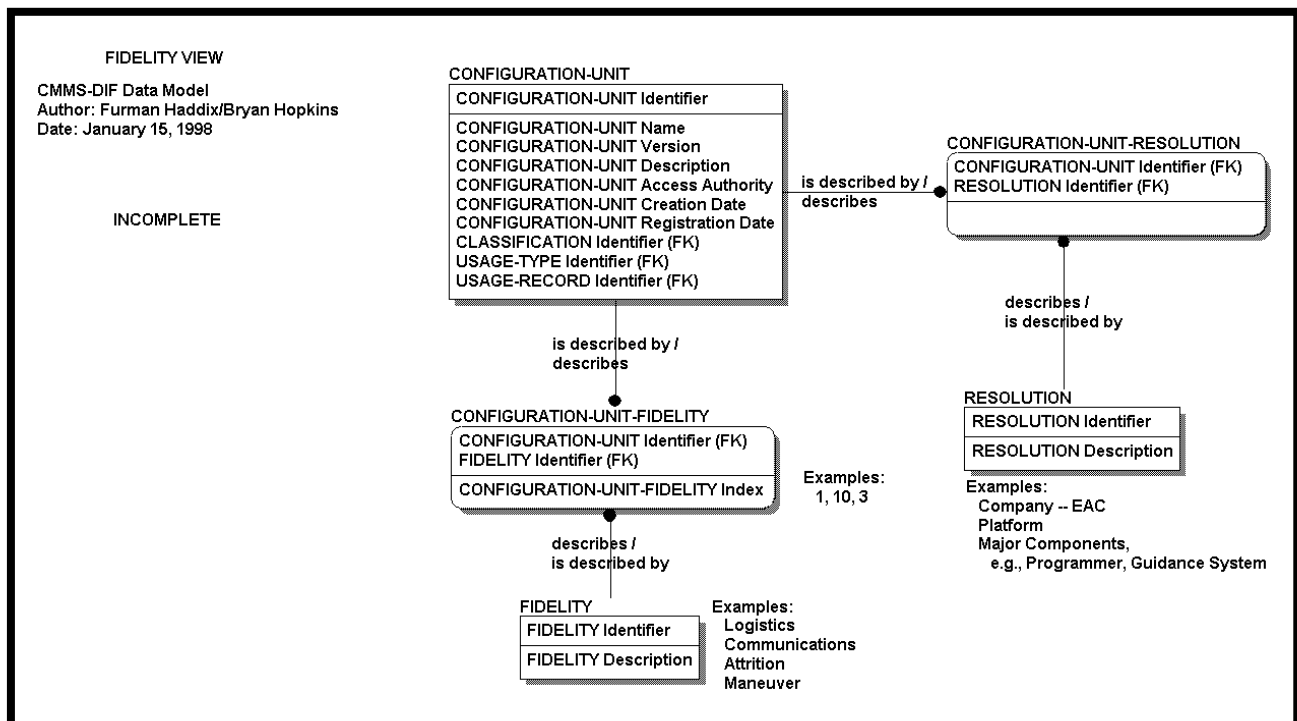
5.21 Fidelity

Fidelity:

Discussion – Fidelity is the degree to which a model represents the real world. The highest fidelity model is an emulation, which is a simulation that captures all aspects of the real world and not only those that are relevant to a certain problem. Fidelity may be viewed as multi-dimensional where dimensions might be attrition, communications, or maneuvering, for example. Each dimension may be characterized separately so that a particular simulation might be high fidelity for communications but low fidelity for maneuvering.

Definition – *Fidelity is the degree to which a model represents the real world.*

The following Fidelity View illustrates Fidelity usage.



PRELIMINARY DRAFT COPY

5.22 State

State:

Discussion – State is a key concept for simulation development, specifically with respect to tasks, which may describe operations of implementation objects. Tasks that do not result in behavior, i.e. change of state or exchange of messages, are of no interest to simulation developers. These are no-ops. The observer (or tester) cannot tell if such a task executed or not. A criterion for a task to be of interest is that it must minimally result in a state change or an interaction. This is one way of telling when you have reached maximum atomicity in task decomposition. If you have only one state change or one interaction as a result of executing your task, then that is maximal atomicity, because if you decompose the task any further, some of its component tasks will be no-ops. Recapping, if a task has no behavior, it is of no interest. Behavior is either a state change or an interaction.

Definition – *A State is one of the possible conditions in which an Object may exist, characterized by definite quantities that are distinct from other quantities; at any given point in time, the state of an object encompasses all of the (usually static) properties of the object plus the current (usually dynamic) values of each of these properties [9].*

State Transition:

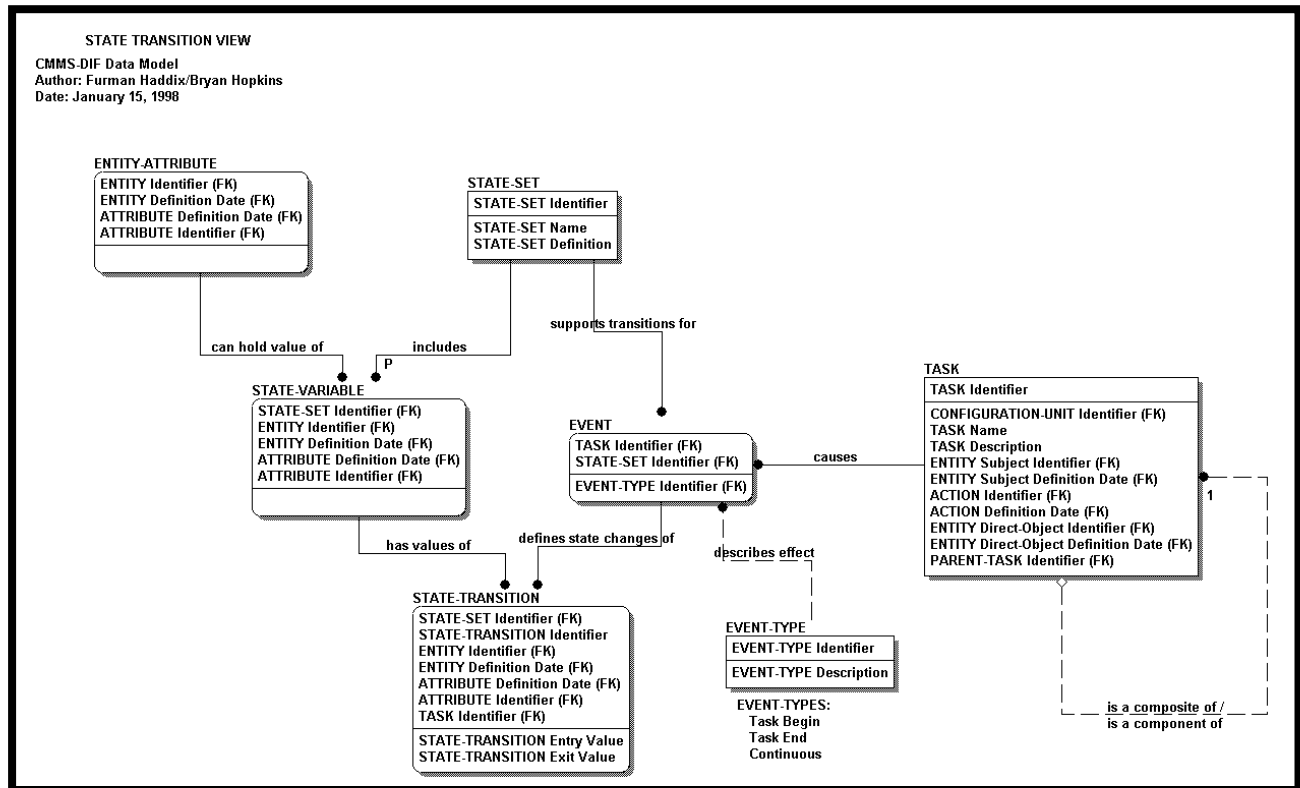
Definition – *A State Transition is a relationship between two States such that an Entity moves from one State to the other when certain conditions are satisfied. [2].*

State Variable:

Definition – *A State Variable describes a property of an object such that the value of the State Variable (in conjunction with the values of other State Variables if any exist for the object) determines the state of the object.*

The following State Transition View illustrates State Transition usage.

PRELIMINARY DRAFT COPY



Appendix A. Glossary

A-1. CMMS Glossary

Some of the following terms are defined from two points of view, conceptual and constructive or technical. In each such case, the conceptual definition is definition (a), and the constructive definition is definition (b).

Accreditation – An accreditation is an official determination that a model or simulation is acceptable for a specific purpose [7].

Action – (a) An action is the alteration or transformation by natural force or human agency that produces an event such as move, sense, communicate, engage, or replenish. (b) An action is a task-template description having a single context-free meaning and consisting of a verb, subject, direct object, and optionally, indirect object phrases. The meaning of the action is tied to a single meaning of its verb if the verb has multiple meanings.

Actor – An actor is a shell that only provides a stimulus. Normally, no attempt is made to characterize its state or behavior, other than that it is a source of inputs or sink of outputs for the behavior of other entities.

Adjacency – Adjacency is the spatial relationship between two entities delineating the possibility of cooperation or attack.

PRELIMINARY DRAFT COPY

Agent – An agent is an entity that is a person or an organization. It is an entity that can be the subject of a use case. A vehicle or platform does not have behavior except as it derives from personal or organizational decisions and/or activities.

Aggregation – Aggregation is the relationship between components and their assembly.

Assignment – An assignment is the representation of an initiation of a use case thread for a subject as an output of a task or activity by an authority entity. An attribute of an assignment is a use case (thread).

Association – An association is a relationship between two entities indicating an area of common interest. Associations include adjacency, collaboration, command, communication, control, occupancy, occupation, operation, opposition, organization, possession, supply, and support.

Attribute – An attribute is information about an entity carried by that entity.

Attack – Attack is the interaction in which projectiles or other potentially damaging materiel are sent from an agent to another entity.

Capability – Capability is an entity's ability to perform a certain action.

Certification – Certification is the approval of a component of CMMS or a set of related components of CMMS for a specific use.

Collaboration – Collaboration is the relationship between agents in which there are communication and some commonality of purpose but where the stronger relationships of command or support are not present.

Collision – Collision is the interaction in which two or more objects attempt to occupy the same space at once.

Command – Command is the relationship between a superior agent and a subordinate agent in which the superior agent may assign tasks, standards, and objectives to the subordinate.

Communication – Communication is the relationship between two entities in which information is exchanged.

Conceptual Object – A conceptual object is a first-order abstraction of a real-world object. It may be a collection of abstractly or concretely connected agents that are organized to accomplish a specific purpose. A conceptual object can be described by one or more models or views, possibly from different viewpoints.

Conceptual Object Model – A conceptual object model is a semantically closed abstraction of a conceptual object.

Condition – A condition is a variable of an operational environment or situation in which an organization, an equipment, or a person is expected to operate that may affect performance [4].

Configuration Unit – A configuration unit represents a set of related objects and object relationships of arbitrary extent and common production, examination, registration, version, and release characteristics.

PRELIMINARY DRAFT COPY

Control – Control is the relationship between two agents in which one agent determines the behavior of the other through frequent interaction.

Data Dictionary – A data dictionary provides a unique set of semantics for each uniquely identified data element within a context.

Delegation – Delegation is the interaction in which command and control of a subordinate agent are assigned from one superior agent to another.

Entity – An entity is a distinguishable person, place, thing, or concept about which information is kept. In particular, entity includes the notions of person, organization, facility, feature, materiel, network, information, and equipment [6].

Entity Type – An entity type is a functional type of entity.

Enumeration – An enumeration of an element gives the set of values that parameters may take on.

Equipment – Equipment is all nonexpendable items needed to outfit or equip an individual or organization [7].

Event – An event is an occurrence that results in behavior such as change of state or an exchange of messages.

Examination – An examination is a test to determine whether or not a component of CMMS or a set of related components of CMMS is acceptable for a specific use.

Facility – A facility is a real property entity consisting of one or more of the following: a building, a structure, a utility system, pavement, and underlying land [7].

Feature – A feature is any object or configuration of ground or water represented on the face of a map or chart [7].

Fidelity – Fidelity is the degree to which a model represents the real world.

Generic Task – A generic task is a task in which participants have been identified by entity type.

Inactive Entity – An inactive entity is an entity that is part of the use case package but that has no volition of its own and is incapable of taking action based on stimulation.

Information – (a) Information is knowledge, data, or facts about particular events or situations sent from one entity to another. (b) Information is an entity that constitutes an information element or component, as information elements can contain other information elements. The purpose is to provide formatting for messages, reports, plans, orders, etc.

Inheritance – Inheritance is the transfer of attributes and behavior from a superclass to a subclass.

Interaction – An interaction is the sending of information or other elements between two tasks or agents. The types of interaction include assignment, attack, collision, delegation, land, launch, resupply, transfer, and transmission.

Land – Land is the interaction that causes an aircraft to return to surface mode from flight in a controlled and procedurally executed manner.

Launch – Launch is the interaction that transitions an entity from surface mode to dynamic flight.

PRELIMINARY DRAFT COPY

Materiel – Materiel is all items (including ships, tanks, self-propelled weapons, aircraft, etc., and related spares, repair parts, and support equipment, but excluding real property, installations, and utilities) necessary to equip, operate, maintain, and support military activities without distinction as to its application for administrative or combat purposes [7].

Measure of Performance – A measure of performance provides a basis for describing varying levels of performance of tasks or missions [4].

Metadata – Metadata is any data about data other than the data instance value.

Model – A model is a semantically closed abstraction of a system [8].

Network – A network is a system of communications.

Occupancy – Occupancy is the relationship between an agent and an inanimate entity in which the agent may be characterized as having the same location as the entity by virtue of being in, on, or under the inanimate entity.

Occupation – Occupation is the relationship between an agent and a political or inanimate entity in which the agent retains or maintains possession of and exercises authority and effective control over the inanimate entity.

Operation – Operation is the relationship between an agent and an inanimate entity in which the agent interacts with the entity in order to moderate the entity's behavior.

Opposition – Opposition is the relationship in which one agent directly or indirectly attempts to prevent the achievement of the objectives of the other agent and vice versa.

Organization – An organization is the association of the agents of a command, including at least a commander and one other person or organization.

Person – A person is a single individual who can be the subject of a use case. A person is a type of agent.

Player – A player is an element that a simulation developer might want to implement in a simulation and is a composite of all the roles that an agent might perform.

Possession – Possession is the relationship between an agent and an inanimate entity that conveys the right of utilization and/or disposition of the entity.

Process – A process is a sequence of actions.

Producer – A producer produces a system. Producer is a type of user.

Reference – A reference provides the authoritative source or sources upon which an element in the model is based.

Resupply – Resupply is the interaction that provides additional expendable materiel.

Role – A role consists of one or more operation specifications and the state variables (attributes) necessary to support them. It is an entity that defines the function provided by, the part played by, or the character assigned to an entity in a process.

Sequence – A sequence defines a temporal relationship between two tasks, actions, or processes.

PRELIMINARY DRAFT COPY

Specific Task – A specific task is a task in which participants have been identified by specific entity.

Standard – A standard provides a way of expressing the degree to which an entity must perform a use case under a specified set of conditions.

State – A state is one of the possible conditions in which an object may exist, characterized by definite quantities that are distinct from other quantities; at any given point in time, the state of an object encompasses all of the (usually static) properties of the object plus the current (usually dynamic) values of each of these properties [9].

State Transition – A state transition is a relationship between two states such that an entity moves from one state to the other when certain conditions are satisfied.

State Variable – A state variable describes a property of an object such that the value of the state variable (in conjunction with the values of other state variables if any exist for the object) determines the state of the object.

Subtask – A subtask is a recursively decomposable component of a task.

Supply – Supply is the relationship between two agents in which one agent provides essentials to the other.

Support – Support is the relationship between agents in which one agent's objective is to facilitate the achievement by the other agent of its objectives.

System – A system is a collection of connected units that are organized to accomplish a specific purpose. A system can be described by one or more models, possibly from different viewpoints [8].

Task – (a) A task is a description of a military activity, including the activity performer and the activity object(s). It may be decomposed into subtasks (recursively decomposable) or steps (atomic). (b) A task is the execution of an action by an entity. The entity initiates execution when specific entrance criteria are met. During execution the task may receive or consume one or more inputs from suppliers, may produce or deliver one or more outputs to receivers, and may change one or more entity attributes. Execution continues until specific exit criteria are satisfied.

Task Conditions – A task condition is a variable in an environment or situation that may affect performance of a task.

Task Step – A task step is an atomic component of a task.

Transfer – A transfer is the interaction used for shift of control over materiel or a facility from one agent to another.

Transmission – Transmission is the interaction in which information or disinformation is sent from one agent to another.

Use Case – (a) A use case is a description of the behavior of an entity (responding entity) as the result of stimulation by another entity (stimulating entity). (b) A use case is a coherent unit of functionality provided by one or more objects.

Use Case Instance – (a) A use case instance is an instance of a use case possibly characterized by instances of physical, military, and civil conditions, a geographic context, orders of

PRELIMINARY DRAFT COPY

battle, and performance standards. (b) A use case instance provides illustrative environment and starting conditions, measures of performance, and assumptions critical to task flow. Use case instances provide a plausible political/strategic and military/operational context and describe the conditions within which battlespace entities interact with each other.

Use Case Package – A use case package is a set of use cases that share a common organizing principle, purpose, or feature. The use cases in a package may be related by function, type, objective, performer, or other relationships.

User – A user is a person or organization that uses a system, whether as examiner, extractor, producer, reader, repository manager, or sponsor.

Validation – Validation is the process of determining the degree to which a model or simulation is an accurate representation of the real world from the perspective of the intended users of the model or simulation [7].

Verb – A verb is a descriptor of an action performed by an entity in the military operations mission space. For the action to be of interest to CMMS, it must result in a change of state in the entity performing the action or in the entity that is the object of the action.

Verification – Verification is the process of determining that a model or simulation implementation accurately represents the developer’s conceptual description and specifications [7].

A-2. Document Usage Glossary

Identity – Identity means that objects are distinguishable by their existence and not by their characteristics.

Object Modeling Technique – Object Modeling Technique (OMT) is a CASE (Computer Assisted Software Engineering) tool based on work of Rumbaugh and others (Rational).

Object-oriented – Different authors cite different characteristics that make a paradigm “object-oriented”. Key concepts often included are the following: identity, inheritance, encapsulation/information hiding, polymorphism, and dynamic binding.

Objectory – Objectory is a methodology based on work of Jacobson and others (Rational).

SOMATiK – SOMATiK is a CASE tool based on SOMA (Semantic Object Modeling Approach) (Bezant Object Technology).

StateMate – StateMate is a CASE tool based on state work done by Harel at Weizmann Institute (I-logix).

Unified Modeling Language – Unified Modeling Language is a CASE tool specification based on combination of Booch, Objectory, and OMT methodologies (Rational).

A-3. Acronyms

BDE – Brigade

BN – Battalion

PRELIMINARY DRAFT COPY

BOM – Business Object Model

CASE – Computer (Aided/Assisted) Software Engineering

CDR – Commander

CJCS – Chairman, Joint Chiefs of Staff

CJCSM – CJCS Manual

CMMS – Conceptual Models of the Mission Space

COM – Conceptual Object Model

CONUS – Continental United States

DMSO – Defense Modeling and Simulation Office

DoD – Department of Defense

DOM – Domain Object Model

GDP – Gross Domestic Product

HNS – Host Nation Support

IOM - Implementation Object Model

JAD – Joint Application Development (Workshop)

JMETL – Joint Mission Essential Task List

JP – Joint Publication

JSIMS – Joint Simulation System

JTM – Joint Training Manual

JTMP – Joint Training Master Plan

JWARS – Joint Warfare (Simulation) System

KDP – Knowledge Development Plan

KL – Knowledge Element

LOC – Lines of Communication

LRC – Lesser Regional Contingency

MOG – Maximum on Ground

MOMS – Military Operations Mission Space

MRC – Major Regional Contingency

MTOT – *Migrating to Object Technology*, Ian Graham, 1994

NCA – National Command Authority

OMG – Object Management Group

OMT – Object Modeling Technique

PRELIMINARY DRAFT COPY

OOM&D – *Object-Oriented Modeling and Design*, Rumbaugh et al., 1991

RAD – Rapid Application Development (Workshop)

ROE – Rules of Engagement

SME – Subject Matter Expert

SOFA – Status of Forces Agreements

SOMA – Semantic Object Modeling Approach

SOMATiK – SOMA Tool Kit

TF – Technical Framework

TOM – Task Object Model

TPFDD – Time-Phased Force and Deployment Data

UJTL – *Universal Joint Task List*, Vn. 3.0 (CJCSM 3500.04A)

UML – Unified Modeling Language

VV&A – Verification, Validation, and Accreditation

VV&C – Verification, Validation, and Certification

WARSIM 2000 – Warfighters Simulation 2000

Appendix B. References

1. James Rumbaugh, et al, *Object-Oriented Modeling and Design*, Prentice Hall, Englewood Cliffs, New Jersey, 1991.
2. “UML Notation Guide,” version 1.1 alpha 6 (1.1 c), Rational Software Corporation, 21 July 1997.
3. Ian Graham, *Migrating to Object Technology*, Addison-Wesley Publishing Co., Reading, Massachusetts, 1995.
4. Universal Joint Task List (UJTL), CJCSM 3500.04A, Version 3.0, 13 September 1996.
5. several JWARS documents
6. Jack Sheehan, et al, “Conceptual Models of the Mission Space (CMMS) Technical Framework, version 0.2.1,” Defense Modeling and Simulation Office, 13 February 1997.
7. DoD Dictionary of Military Terms and Associated Terms, JP 1-02, 23 March 1994.
8. “UML Semantics,” version 1.0, Rational Software Corporation, 13 January 1997.
9. Grady Booch, *Object-Oriented Analysis and Design with Applications*, 2nd ed., Benjamin/Cummings Publishing Co., Redwood City, California, 1994.

Appendix C. Environment

In section 5.1.9 (“Task Conditions”) we discussed conditions that may affect the performance of a task. These conditions may be divided into three major categories: conditions

PRELIMINARY DRAFT COPY

in the physical environment, conditions in the military environment, and conditions in the civil environment. Conditions affect tasks differently, and a condition that has a great effect on the performance of one task may have an insignificant effect or no effect at all on the performance of another. Moreover, conditions may affect friendly and hostile forces differently. For example, conditions in the physical environment are likely to affect friendly and hostile forces in roughly the same way since both must deal with the climate, weather, and terrain of a given area. However, conditions in the military environment, such as those having to do with forces or firepower, may differ for friendly and hostile forces. Similarly, conditions in the civil environment, such as those having to do with political support, culture, or economy, may differ for friendly and hostile forces.

C-1. Physical Environment

The physical environment includes not only the natural environment but also changes made to the natural environment by civilization. The physical environment may be divided into the following domains: land, sea, air, and space. The Chairman of the Joint Chiefs of Staff Manual Universal Joint Task List (CJCSM UJTL) [4] gives definitions and descriptors of these conditions on pp. 3-5 to 3-14.

C-1.1. Land

The section “Conditions for Joint Tasks” in the UJTL, which gives a complete list of the environmental conditions that may be used to describe the conditions under which tasks are performed, defines the domain land as “Physical characteristics, both natural and synthetic, of a land area” [4]. Under the domain land are the following subcategories: terrain, geological features, synthetic terrain features, and landlocked waters. Under the subcategory terrain are the following conditions: terrain relief, terrain elevation, terrain slope, terrain firmness, terrain traction, vegetation, and terrain relief features. Under the subcategory geological features are the following conditions: geological activity, magnetic variation, and subsurface water. Under the subcategory synthetic terrain features are the following conditions: urbanization, significant civil structures, synthetic terrain contrast, obstacles to movement, and route availability. Under the subcategory landlocked waters are the following conditions: landlocked waters depth, landlocked waters currents, landlocked waters width, landlocked waters bottom, and landlocked waters shore gradient.

C-1.2. Sea

The UJTL “Conditions for Joint Tasks” defines the domain sea as “Those factors associated with the continuous salt water ocean system to include oceans, seas, gulfs, inlets, bays, sounds, straits, channels, and rivers” [4]. Under the domain sea are the following subcategories: ocean waters, ocean bottom, harbor capacity, littoral characteristics, riverine environment, and shipping presence. Under the subcategory ocean waters are the following conditions: ocean depth, ocean currents, sea state, ocean temperature, saline content, ocean features, sea room, ocean acoustics, ocean bioluminescence, ocean ice, ocean ice thickness, and ocean ambient noise. Under the subcategory ocean bottom are the following conditions: sea bottom contours and sea bottom composition. Under the subcategory harbor capacity are the following conditions: harbor shelter, harbor depth, and harbor currents. Under the subcategory littoral characteristics are the

PRELIMINARY DRAFT COPY

following conditions: littoral gradient, littoral composition, littoral terrain features, littoral tides, and littoral currents. Under the subcategory riverine environment are the following conditions: riverine navigability, riverine tidal turbulence, riverine current, and riverine bank gradient. Under the subcategory shipping presence are the following conditions: shipping density, shipping type, and shipping identifiability.

C-1.3. Air

The UJTL “Conditions for Joint Tasks” defines the domain air as “Characteristics of the lower atmosphere, to include climate, visibility, and weapons effects” [4]. Under the domain air are the following subcategories: climate, visibility, atmospheric weapon effects, and airspace availability. Under the subcategory climate are the following conditions: season, weather systems, and weather (which includes air temperature, barometric pressure, surface wind velocity, low altitude wind velocity, medium altitude wind velocity, high altitude wind velocity, wind direction, humidity, precipitation, and altitude). Under the subcategory visibility are the following conditions: light and obscurants. Under the subcategory atmospheric weapon effects are the following conditions: nuclear effects (which includes nuclear blast/thermal effects and nuclear radiation effects), chemical effects, biological effects, and electromagnetic effects. Under the subcategory airspace availability are no further conditions.

C-1.3.1 Weather

Adequate handling of weather will require regionalization. Two ways of handling this are the following.

First, weather may be defined for regions. When an entity moves from one region to another, weather conditions change.

Second, a more sophisticated method is to define weather conditions at points. Then interpolation can be performed based on the closest two, three, or more points. Multi-point interpolation can be based on the following formulae.

$$W_i = 1/D_i \quad // \text{ Point Weight is inverse of Distance from object to point.}$$
$$M = \sum (W_i * M_i) / \sum W_i \quad // \text{ Measure = Sum of Point Weight-Point Measure Products}$$
$$\quad // \text{ Divided by Sum of Point Weights.}$$

Example:

Five Points with Temperature—

P1: D, 1 km; M, 84° F.
P2: D, 100 km; M, 78° F.
P3: D, 42 km; M, 81° F.
P4: D, 58 km; M, 76° F.
P5: D, 12 km; M, 76° F.

Computations—

P1: W, 1.00; WT, 84.0.
P2: W, .010; WT, 0.8.
P3: W, .024; WT, 1.9.
P4: W, .017; WT, 1.3.
P5: W, .083; WT, 6.3.
Total: $\sum W_i$, 1.134; $\sum W_i M_i$, 94.3.
M: 83.2° F.

PRELIMINARY DRAFT COPY

Still more complex is to consider altitudes as well.

C-1.4. Space

The UJTL “Conditions for Joint Tasks” defines the domain space as “Characteristics of the upper reaches of Earth’s atmosphere” [4]. Under the domain space are the following subcategories: objects in space, solar and geomagnetic activity, and high energy particles. Under the subcategory objects in space are the following conditions: orbit density and orbit type. Under the subcategories solar and geomagnetic activity and high energy particles are no further conditions.

C-2. Military Environment

The military environment has to do with conditions related to military forces, whether US, friendly, neutral, or enemy. The military environment may be divided into the following domains: mission; forces; command, control, and communications; intelligence; deployment, movement, and maneuver; firepower; protection; sustainment; and threat. The UJTL gives definitions and descriptors of these conditions on pp. 3-15 to 3-30.

C-2.1. Mission

The UJTL “Conditions for Joint Tasks” defines the domain mission as “Those factors that frame and influence the execution of the mission assigned or understood” [4]. Under the domain mission are the following subcategories: mission instructions, legal state, mission preparation, theater dimensions, and time available. Under the subcategory mission instructions are the following conditions: command level, pre-existing arrangements, mission classification, ROE (Rules of Engagement), SOFA (Status of Forces Agreements), military commitments to other nations, and military commitments from other nations. Under the subcategories legal state and mission preparation are no further conditions. Under the subcategory theater dimensions are the following conditions: location, theater(s), joint operations area, tactical area of responsibility, intertheater distance, and intratheater distance. Under the subcategory time available are the following conditions: lead time and mission duration.

C-2.2. Forces

The UJTL “Conditions for Joint Tasks” defines the domain forces as “The overall capabilities of the forces of a nation, alliance, or coalition” [4]. Under the domain forces are the following subcategories: forces assigned, competing apportionments, forces allocated, personnel capability, modern military systems, interoperability, and military force relationships. Under the subcategories forces assigned, competing apportionments, and forces allocated are no further conditions. Under the subcategory personnel capability are the following conditions: personnel nutrition and health, personnel literacy, personnel physical conditioning, personnel morale, personnel experience, and personnel fatigue. Under the subcategory modern military systems are the following conditions: modern weapons systems, modern information and intelligence processing systems, military systems reliability, and military systems maturity. Under the subcategories interoperability and military force relationships are no further conditions.

PRELIMINARY DRAFT COPY

C-2.3. Command, Control, and Communications

The domain command, control, and communications has to do with management of forces and with communication among staff and forces. Under the domain command, control, and communications are the following subcategories: command arrangements and military style. Under the subcategory command arrangements are the following conditions: joint staff integration, multinational integration, staff expertise, pre-existing command, command authority, communications connectivity, classification, information exchange, and information volume. Under the subcategory military style are the following conditions: leadership style, force emphasis, flexibility of warfare style, and component headquarters location.

C-2.4. Intelligence

The domain intelligence has to do with data and information. Under the domain intelligence are the following conditions: warning, intelligence data base, theater intelligence organization, theater intelligence access, intelligence countermeasure capability, and certitude of data.

C-2.5. Deployment, Movement, and Maneuver

The domain deployment, movement, and maneuver has to do with the deployment and maneuver of forces and the movement of forces or materiel. Under the domain deployment, movement, and maneuver are the following subcategories: LOC (lines of communication) and planning status, lift assets, en route support, and reception and onward movement. Under the subcategory LOC and planning status are the following conditions: TPFDD (time-phased force and deployment data) availability, deployment lead time, intertheater LOCs, intratheater LOCs, and entry capability. Under the subcategory lift assets are the following conditions: airlift assets, sealift assets, ground transportation assets, spacelift assets, and refueling assets. Under the subcategory en route support are the following conditions: intermediate staging bases, overflight/passage rights, and en route supply. Under the subcategory reception and onward movement are the following conditions: reception facilities (which includes wharfage, maximum on ground (MOG), runway length, and runway weight bearing capacity) and onward movement facilities (which includes beddown facilities, marshaling facilities, and staging area).

C-2.6. Firepower

The domain firepower has to do with the ability to deliver firepower to targets. Under the domain firepower are the following conditions: degree of dispersion, degree of camouflage, target hardness, preplanned targets, target mobility, target range, collateral damage potential, and target thermal contrast.

C-2.7. Protection

The domain protection has to do with area security and arena superiority. Under the domain protection are the subcategories rear area/local security, air superiority, space control, maritime superiority, and ground superiority. Under the subcategories rear area/local security and air superiority are no further conditions. Under the subcategory space control are the following conditions: space platforms, space platforms (availability), and space platforms (linkability). Under the subcategories maritime superiority and ground superiority are no further conditions.

PRELIMINARY DRAFT COPY

C-2.8. Sustainment

The domain sustainment has to do with materiel and services necessary for the sustaining of forces. Under the domain sustainment are the following conditions: sustainment facilities, deployed supplies, CONUS (Continental United States) resupply, pre-positioned materiel, host nation support (HNS), and commercial procurement.

C-2.9. Threat

The domain threat has to do with the existence and nature of military threats. Under the domain threat are the following subcategories: threat (seriousness of to the nation), threat form, threat existence, threat posture, threat size, and threat disposition. Under the subcategories threat (seriousness of to the nation), threat form, threat existence, and threat posture are no further conditions. Under the subcategory threat size are the following conditions: threat land force size, threat naval force size, and threat air force size. Under the subcategory threat disposition are no further conditions.

C-3. Civil Environment

The UJTL “Conditions for Joint Tasks” defines the civil environment as “Those factors related to a people, their government, politics, culture, and economy that impact military operations” [4]. The civil environment may be divided into the following domains: political policies, culture, and economy. The UJTL gives definitions and descriptors of these conditions on pp. 3-31 to 3-40.

C-3.1. Political Policies

The UJTL “Conditions for Joint Tasks” defines the domain political policies as “Those factors that derive from the people, their national government, and international and non-government organizations that support or oppose military action” [4]. Under the domain political policies are the following subcategories: domestic political support, international politics, and NCA (National Command Authority) decisions. Under the subcategory domestic political support are the following conditions: domestic public support, congressional support, interdepartmental/interagency relationships, legality, and press relations. Under the subcategory international politics are the following conditions: major power involvement, foreign government stability, foreign government support, foreign public opinion, international organization support, and multinational business support. Under the subcategory NCA decisions are the following conditions: number of crises, mission priority, mobilization level (which includes force level, draft, and mobilization facilities), and restraints on action.

C-3.2. Culture

The UJTL defines the domain culture as “Those aspects of a people that relate to their language, customs, economics, religion, and character” [4]. Under the domain culture are the following subcategories: language, customs adjustment, religious beliefs, significant cultural sites, cultural unity, and national character. Under the subcategory language are the following conditions: language translation and language translators. Under the subcategory customs adjustment are the following conditions: societal openness, legal penalties, and law source. Under the subcategory religious beliefs are the following conditions: religious unity, religious militancy, and religion-state relationship. Under the subcategories significant cultural sites and cultural unity are no

PRELIMINARY DRAFT COPY

further conditions. Under the subcategory national character are the following conditions: national discipline, national aggressiveness, nationalism, ethnocentrism, and internationalism.

C-3.3. Economy

The UJTL defines the domain economy as “Those factors that provide a nation with the manpower, materiel, and money to allow it to play a role on the military stage and shape that role” [4]. Under the domain economy are the following subcategories: population, refugee impact, gross domestic product (GDP), international economic position, industry, national potential, and science and technology. Under the subcategory population are the following conditions: size of military, population growth rate, educated population, civil health, health risk, and civil unrest. Under the subcategory refugee impact are the following conditions: refugee (type), refugee congestion, refugee care responsibility, and refugee relocation effort. Under the subcategory gross domestic product (GDP) are no further conditions. Under the subcategory international economic position are the following conditions: economic self-sufficiency (which includes self-sufficiency in food, self-sufficiency in fuel, self-sufficiency in raw materials, self-sufficiency in finished goods, and self-sufficiency in machinery), fiscal position, and infrastructure dependence. Under the subcategory industry are the following conditions: industrialization, industrial growth rate, electrical production, and armaments production capacity. Under the subcategory national potential are the following conditions: transportation infrastructure, telecommunications infrastructure, and available capital. Under the subcategory science and technology are the following conditions: basic research, research application (military), high technology production, and information management.

Appendix D. Entity Examples

An entity is a distinguishable person, place, thing, or concept about which information is kept [6]. It is a conceptual object having battlefield relevance. Usually, it will clearly represent one or more aspects of a real world object.

Entity (root)

Agent

Organization

Friendly

Division

Armor

Mechanized Infantry

Brigade

Battalion

Company

Platoon

Squad

Opposing

Civilian

Individual

Role

Position

PRELIMINARY DRAFT COPY

Rank (Organization/Rank or Civilian)

Other

Materiel

Ammunition

Equipment

Reports

Plans

Events

Assessments

Feature

fault

river

mountain

pond

marsh

forest

Facility

fortification

village

bridge

highway

Weather Region (10 km squares, 100 km squares??)

Appendix E. Related Data Structures

This section gives related data structures for certain concepts in CMMS.

E-1. Action

Related Data Structures:

Action:

Action Identifier (FK):

Action Definition Date (FK):

Subject Identifier (FK):

Subject (FK):

Verb Identifier (FK):

Verb Definition Date (FK):

Direct-Object Identifier (FK):

Direct-Object Definition Date (FK):

PRELIMINARY DRAFT COPY

Verb:

Verb Identifier (FK):

Verb Definition Date (FK):

Verb Kind-of Definition Date (FK):

Verb Identifier (FK):

Synonym Definition Date (FK):

Phrase:

Phrase Sequence Identifier:

Action Definition Date (FK):

Action Identifier (FK):

Entity Identifier (FK):

Entity Definition Date (FK):

Modifier Identifier (FK):

Modifier:

Modifier Identifier:

Modifier Name:

Modifier Definition:

{ And, For, From, Of, Or, To, Using, With }

Entity:

Entity Identifier (FK):

Entity Definition Date (FK):

Entity Kind-of Definition Date (FK):

Entity Kind-of Identifier (FK):

Entity Identifier (FK):

Entity Definition Date (FK):

PRELIMINARY DRAFT COPY

Task:

Task Identifier:

Configuration-Unit Identifier (FK):

Task Name:

Task Description:

Entity Subject Identifier (FK):

Entity Subject Definition Date (FK):

Action Identifier (FK):

Action Definition Date (FK):

Entity Direct-Object Identifier (FK):

Entity Direct-Object Definition Date (FK):

Parent-Task Identifier (FK):

E-2. Action Sequence

Related Data Structures:

Action-Sequence:

Second-Action Identifier (FK):

Process-Package Identifier (FK):

Process Identifier (FK):

First-Part Identifier (FK):

Action Identifier (FK):

Action Definition Date (FK):

Second-Action Definition Date (FK):

Second-Part Identifier (FK):

Action-Sequence First Part Cardinality:

Action-Sequence Second Part Cardinality:

PRELIMINARY DRAFT COPY

Action-Sequence Iteration Description:

Process:

Process-Package Identifier (FK):

Process Identifier:

Configuration-Unit Identifier (FK):

Process Name:

Process Description:

Action:

Action Identifier (FK):

Action Definition Date (FK):

Subject Identifier (FK):

Subject (FK):

Verb Identifier (FK):

Verb Definition Date (FK):

Direct-Object Identifier (FK):

Direct-Object Definition Date (FK):

First-Part:

First-Part Identifier:

First-Part Name:

First-Part Definition:

{Start, End, Iteration, Concurrent}

Second-Part:

Second-Part Identifier:

Second-Part Name:

Second-Part Description:

PRELIMINARY DRAFT COPY

{Prior, Start, During, End, After}

E-3. Association

Related Data Structures:

Association:

Association-Stereotype Identifier (FK):

Association Cardinality:

Entity-Association:

Association-Stereotype Identifier (FK):

Entity Definition Date (FK):

Entity Identifier (FK):

Association Cardinality:

Entity:

Entity Identifier (FK):

Entity Definition Date (FK):

Entity Kind-of Definition Date (FK):

Entity Kind-of Identifier (FK):

Entity Identifier (FK):

Entity Definition Date (FK):

Association-Stereotype:

Association-Stereotype Identifier:

Association-Stereotype Name:

Association-Stereotype Description:

{Adjacency (Is Adjacent To), Collaboration (Collaborates With), Command (Commands), Communication (Communicates With), Control (Controls), Occupancy (Occupies), Occupation (Occupies), Operation (Operates), Opposition (Opposes), Organization (Is Group Of), Possession (Owns), Supply (Supplies), Support (Supports)}

PRELIMINARY DRAFT COPY

Use-Case-Package:

Use-Case-Package Identifier:

Configuration-Unit Identifier (FK):

Use-Case-Package Name:

Use-Case-Package Description:

Use-Case-Package Containing Identifier (FK):

E-4. Capability

Related Data Structures:

Capability:

Entity Identifier (FK):

Entity Definition Date (FK):

Activity Identifier (FK):

Activity Definition Date (FK):

Entity:

Entity Identifier (FK):

Entity Definition Date (FK):

Entity Kind-of Definition Date (FK):

Entity Kind-of Identifier (FK):

Entity Identifier (FK):

Entity Definition Date (FK):

Action:

Action Identifier (FK):

Action Definition Date (FK):

Subject Identifier (FK):

Subject (FK):

PRELIMINARY DRAFT COPY

Verb Identifier (FK):

Verb Definition Date (FK):

Direct-Object Identifier (FK):

Direct-Object Definition Date (FK):

E-5. Condition

Related Data Structures:

Use-Case-Condition:

Use-Case-Package Identifier (FK):

Use-Case Identifier (FK):

Use-Case-Condition Identifier:

Condition Identifier (FK):

Entity Identifier (FK):

Entity Definition Date (FK):

Attribute Identifier (FK):

Attribute Definition Date (FK):

Relation Identifier (FK):

Entity-Attribute:

Entity Identifier (FK):

Entity Definition Date (FK):

Attribute Definition Date (FK):

Attribute Identifier (FK):

Use-Case:

Use-Case Identifier:

Use-Case-Package Identifier (FK):

Entity Stimulating Identifier (FK):

PRELIMINARY DRAFT COPY

Entity Stimulating Definition Date (FK):

Entity Responding Identifier (FK):

Entity Responding Definition Date (FK):

Use-Case Name:

Use-Case Purpose:

Use-Case Description:

Configuration-Unit Identifier (FK):

Relation:

Relation Identifier:

Relation Name:

Relation Definition:

{ Equals, Greater Than, Greater Than or Equals, Less Than, Less Than or Equals, Not Equal }

Condition-Status:

Use-Case-Package Identifier (FK):

Use-Case Identifier (FK):

Use-Case-Instance Identifier (FK):

Use-Case-Condition Identifier (FK):

Condition-Status Value:

E-6. Configuration Unit

Related Data Structures:

Configuration-Unit:

Configuration-Unit Identifier:

Configuration-Unit Name:

Configuration-Unit Version:

PRELIMINARY DRAFT COPY

Configuration-Unit Description:

Configuration-Unit Access Authority:

Configuration-Unit Creation Date:

Configuration-Unit Registration Date:

Classification Identifier (FK):

Usage-Type Identifier (FK):

Usage-Record Identifier (FK):

Use-Case-Package:

Use-Case-Package Identifier:

Configuration-Unit Identifier (FK):

Use-Case-Package Name:

Use-Case-Package Description:

Use-Case-Package Containing Identifier (FK):

Use-Case:

Use-Case Identifier:

Use-Case-Package Identifier (FK):

Entity Stimulating Identifier (FK):

Entity Stimulating Definition Date (FK):

Entity Responding Identifier (FK):

Entity Responding Definition Date (FK):

Use-Case Name:

Use-Case Purpose:

Use-Case Description:

Configuration-Unit Identifier (FK):

Use-Case-Instance:

PRELIMINARY DRAFT COPY

Use-Case-Package Identifier (FK):

Use-Case Identifier (FK):

Use-Case-Instance Identifier:

Configuration-Unit Identifier (FK):

Use-Case-Instance Name:

Use-Case-Instance Description:

Task:

Task Identifier:

Configuration-Unit Identifier (FK):

Task Name:

Task Description:

Entity Subject Identifier (FK):

Entity Subject Definition Date (FK):

Action Identifier (FK):

Action Definition Date (FK):

Entity Direct-Object Identifier (FK):

Entity Direct-Object Definition Date (FK):

Parent-Task Identifier (FK):

Process-Package:

Process-Package Identifier:

Configuration-Unit Identifier (FK):

Process-Package Name:

Process-Package Description:

Process:

Process-Package Identifier (FK):

PRELIMINARY DRAFT COPY

Process Identifier:

Configuration-Unit Identifier (FK):

Process Name:

Process Description:

E-7. Data Dictionary

Related Data Structures:

Data-Dictionary:

Data-Dictionary Identifier:

Data-Dictionary Name:

Data-Dictionary Description:

{ Action, Attribute, Entity, Enumeration, Verb }

Data-Dictionary-Release:

Data-Dictionary Identifier (FK):

Data-Dictionary-Release Date:

Data-Dictionary-Release Description:

Attribute-Data-Dictionary:

Data-Dictionary Identifier (FK):

Entity-Data-Dictionary:

Data-Dictionary Identifier (FK):

Activity-Data-Dictionary:

Data-Dictionary Identifier (FK):

Verb-Data-Dictionary:

Data-Dictionary Identifier (FK):

Enumeration-Data-Dictionary:

Data-Dictionary Identifier (FK):

PRELIMINARY DRAFT COPY

Data-Dictionary-Element:

Data-Dictionary-Element Identifier:
Data-Dictionary-Element Definition Date:
Data-Dictionary Identifier (FK):
Data-Dictionary-Element Name:
Data-Dictionary-Element Definition:
Data-Dictionary-Element Definition Authority:
Data-Dictionary-Element Type:
Data-Dictionary-Element-Sponsor Identifier (FK):

Attribute:

Attribute Identifier (FK):
Attribute Definition Date (FK):
Attribute Kind-of Definition Date (FK):
Attribute Identifier (FK):
Attribute Definition Date (FK):

Entity:

Entity Identifier (FK):
Entity Definition Date (FK):
Entity Kind-of Definition Date (FK):
Entity Kind-of Identifier (FK):
Entity Identifier (FK):
Entity Definition Date (FK):

Action:

Action Identifier (FK):
Action Definition Date (FK):

PRELIMINARY DRAFT COPY

Subject Identifier (FK):

Subject (FK):

Verb Identifier (FK):

Verb Definition Date (FK):

Direct-Object Identifier (FK):

Direct-Object Definition Date (FK):

Verb:

Verb Identifier (FK):

Verb Definition Date (FK):

Verb Kind-of Definition Date (FK):

Verb Identifier (FK):

Synonym Definition Date (FK):

User:

User Identifier:

User Name:

User Description:

User POC:

User Voice:

User FAX:

User Electronic:

User Pager:

User Cellular:

User USPS Line 1:

User USPS Line 2:

User USPS Line 3:

PRELIMINARY DRAFT COPY

User USPS City:

User USPS State:

User USPS Zip:

User Other Line 1:

User Other Line 2:

User Other Line 3:

User Other City:

User Other State:

User Other Zip:

E-8. Entity

Related Data Structures:

Entity:

Entity Identifier (FK):

Entity Definition Date (FK):

Entity Kind-of Definition Date (FK):

Entity Kind-of Identifier (FK):

Entity Identifier (FK):

Entity Definition Date (FK):

Data-Dictionary-Element:

Data-Dictionary-Element Identifier:

Data-Dictionary-Element Definition Date:

Data-Dictionary Identifier (FK):

Data-Dictionary-Element Name:

Data-Dictionary-Element Definition:

Data-Dictionary-Element Definition Authority:

PRELIMINARY DRAFT COPY

Data-Dictionary-Element Type:

Data-Dictionary-Element-Sponsor Identifier (FK):

Entity-Attribute:

Entity Identifier (FK):

Entity Definition Date (FK):

Attribute Definition Date (FK):

Attribute Identifier (FK):

Attribute:

Attribute Identifier (FK):

Attribute Definition Date (FK):

Attribute Kind-of Definition Date (FK):

Attribute Identifier (FK):

Attribute Definition Date (FK):

Entity-Entity:

Entity Definition Date (FK):

Entity Identifier (FK):

Entity-Entity-Type Identifier (FK):

Entity-Entity Component Cardinality:

Entity-Entity-Type:

Entity-Entity-Type Identifier:

Entity-Entity-Type Description:

{ OrgOrg – Organization/Organization, OrgPer – Organization/Person, OrgEqp – Organization/Equipment, EqpEqp – Equipment/Equipment, PerEqp – Person/Equipment }

E-9. Entity Type

Related Data Structures:

PRELIMINARY DRAFT COPY

Entity:

Entity Identifier (FK):

Entity Definition Date (FK):

Entity Kind-of Definition Date (FK):

Entity Kind-of Identifier (FK):

Entity Identifier (FK):

Entity Definition Date (FK):

Data-Dictionary-Element:

Data-Dictionary-Element Identifier:

Data-Dictionary-Element Definition Date:

Data-Dictionary Identifier (FK):

Data-Dictionary-Element Name:

Data-Dictionary-Element Definition:

Data-Dictionary-Element Definition Authority:

Data-Dictionary-Element Type:

Data-Dictionary-Element-Sponsor Identifier (FK):

Information:

Information Definition Date (FK):

Information Identifier (FK):

Network:

Network Definition Date (FK):

Network Identifier (FK):

Materiel:

Materiel Definition Date (FK):

Materiel Identifier (FK):

PRELIMINARY DRAFT COPY

Equipment:

Equipment Definition Date (FK):

Equipment Identifier (FK):

Feature:

Feature Definition Date (FK):

Feature Identifier (FK):

Facility:

Facility Definition Date (FK):

Facility Identifier (FK):

Agent:

Agent Definition Date (FK):

Agent Identifier (FK):

Agent-Type Identifier (FK):

Agent-Type:

Agent-Type Identifier:

Agent-Type Description:

{Organization, Person}

Organization:

Organization Definition Date (FK):

Organization Identifier (FK):

Person:

Person Definition Date (FK):

Person Identifier (FK):

E-10. Enumeration

Related Data Structures:

PRELIMINARY DRAFT COPY

Data-Dictionary:

Data-Dictionary Identifier:

Data-Dictionary Name:

Data-Dictionary Description:

{ Activity, Attribute, Entity, Enumeration, Verb }

Data-Dictionary-Release:

Data-Dictionary Identifier (FK):

Data-Dictionary-Release Date:

Data-Dictionary-Release Description:

Enumeration-Data-Dictionary:

Data-Dictionary Identifier (FK):

Usage-Type:

Usage-Type Identifier:

Usage-Type Name:

Usage-Type Definition:

{ Unlimited, No Expiry, Time Period, Per Access List, Per Access, ... }

Operator:

Operator Identifier:

Operator Name:

Operator Definition:

{ Add, Subtract, Multiply, Divide, Percent, Modulo, Negate }

Examination-Type:

Examination-Type Identifier:

Examination-Type Definition:

{ Verification, Validation, Certification }

PRELIMINARY DRAFT COPY

Modifier:

Modifier Identifier:

Modifier Name:

Modifier Definition:

{ And, For, From, Of, Or, To, Using, With }

Usage-Record:

Usage-Record Identifier:

Usage-Record Name:

Usage-Record Definition:

{ Per Usage, Daily Summary by User, Monthly Summary by User, Monthly Hit Report, None }

User-Type:

User-Type Identifier:

User-Type Definition:

{ Examiner, Extractor, Producer, Reader, Repository Manager, Sponsor }

Unit-of-Measure:

Unit-of-Measure Identifier:

Unit-of-Measure Name:

Unit-of-Measure Description:

{ Hours, Minutes, Seconds, Miles, Feet, Kilometers, Meters, Feet per Second, Meters per Second, Percent Successful Identification, Percent Successful Attack, ... }

Classification:

Classification Identifier:

Classification Name:

Classification Description:

PRELIMINARY DRAFT COPY

{Unclassified, Confidential, Secret, Top Secret, No Foreign, NATO}

Association-Stereotype:

Association-Stereotype Identifier:

Association-Stereotype Name:

Association-Stereotype Description:

{Adjacency (Is Adjacent To), Collaboration (Collaborates With), Command (Commands), Communication (Communicates With), Control (Controls), Occupancy (Occupies), Occupation (Occupies), Operation (Operates), Opposition (Opposes), Organization (Is Group Of), Possession (Owns), Supply (Supplies), Support (Supports)}

Interaction-Stereotype:

Interaction-Stereotype Identifier:

Interaction-Stereotype Name:

Interaction-Stereotype Definition:

{Assignment (Initiate mission/task sequence for subordinate), Attack (Send something that will cause damage or casualties), Collision (Attempt to occupy same location), Delegation (Shift command of subordinate unit), Land (Cause to return to surface mode from flight), Launch (Transition an entity from surface mode to dynamic flight), Resupply (Provide additional supply of expendable materiel), Transfer (Shift control of materiel or facility), Transmission (Send information or disinformation)}

Relation:

Relation Identifier:

Relation Name:

Relation Definition:

{Equals, Greater Than, Greater Than or Equals, Less Than, Less Than or Equals, Not Equal}

First-Part:

First-Part Identifier:

First-Part Name:

PRELIMINARY DRAFT COPY

First-Part Definition:

{Start, End, Iteration, Concurrent}

Second-Part:

Second-Part Identifier:

Second-Part Name:

Second-Part Description:

{Prior, Start, During, End, After}

Condition-Clause-Type:

Condition-Clause-Type Identifier:

Condition-Clause-Type Name:

Condition-Clause-Type Definition:

{If, When, While}

Condition-Clause-Kind:

Condition-Clause-Kind Identifier:

Condition-Clause-Kind Name:

Condition-Clause-Kind Description:

{Start, Terminate, Interrupt}

Entity-Entity-Type:

Entity-Entity-Type Identifier:

Entity-Entity-Type Description:

{OrgOrg – Organization/Organization, OrgPer – Organization/Person, OrgEqp – Organization/Equipment, EqpEqp – Equipment/Equipment, PerEqp – Person/Equipment}

E-11. Examination

Related Data Structures:

Examination:

PRELIMINARY DRAFT COPY

Examination Date:

Configuration-Unit Identifier (FK):

Examination-Type Identifier (FK):

Examiner Identifier (FK):

Sponsor Identifier (FK):

Examination-Sponsor Identifier:

Configuration Unit:

Configuration-Unit Identifier:

Configuration-Unit Name:

Configuration-Unit Version:

Configuration-Unit Description:

Configuration-Unit Access Authority:

Configuration-Unit Creation Date:

Configuration-Unit Registration Date:

Classification Identifier (FK):

Usage-Type Identifier (FK):

Usage-Record Identifier (FK):

User:

User Identifier:

User Name:

User Description:

User POC:

User Voice:

User FAX:

User Electronic:

PRELIMINARY DRAFT COPY

User Pager:

User Cellular:

User USPS Line 1:

User USPS Line 2:

User USPS Line 3:

User USPS City:

User USPS State:

User USPS Zip:

User Other Line 1:

User Other Line 2:

User Other Line 3:

User Other City:

User Other State:

User Other Zip:

Examination-Type:

Examination-Type Identifier:

Examination-Type Definition:

{ Verification, Validation, Certification }

System-Accreditation:

Examination Date (FK):

Configuration-Unit Identifier (FK):

System-Accreditation Name:

System-Accreditation Description:

System-Accreditation Use Name:

System-Accreditation Use Description:

PRELIMINARY DRAFT COPY

System-Accreditation System:

Verification:

Examination Date (FK):

Configuration-Unit Identifier (FK):

Verification Name:

Verification Description:

Verification Specification Name:

Verification Specification Description:

Validation:

Examination Date (FK):

Configuration-Unit Identifier (FK):

Validation Name:

Validation Description:

Validation Benchmark Name:

Validation Benchmark Description:

Certification:

Examination Date (FK):

Configuration-Unit Identifier (FK):

Certification Name:

Certification Description:

Certification Use Name:

Certification Use Description:

Verification Date (FK):

Validation Date (FK):

PRELIMINARY DRAFT COPY

E-12. Fidelity

Related Data Structures:

Fidelity:

Fidelity Identifier:

Fidelity Description:

{Examples: Logistics, Communications, Attrition, Maneuver}

Configuration-Unit-Fidelity:

Configuration-Unit Identifier (FK):

Fidelity Identifier (FK):

Configuration-Unit-Fidelity Index:

{Examples: 1, 10, 3}

Configuration-Unit:

Configuration-Unit Identifier:

Configuration-Unit Name:

Configuration-Unit Version:

Configuration-Unit Description:

Configuration-Unit Access Authority:

Configuration-Unit Creation Date:

Configuration-Unit Registration Date:

Classification Identifier (FK):

Usage-Type Identifier (FK):

Usage-Record Identifier (FK):

Configuration-Unit-Resolution:

Configuration-Unit Identifier (FK):

Resolution Identifier (FK):

PRELIMINARY DRAFT COPY

Resolution:

Resolution Identifier:

Resolution Description:

{Examples: Company – EAC, Platform, Major Components (e.g., Programmer, Guidance System)}

E-13. Information

Related Data Structures:

Information:

Information Definition Date (FK):

Information Identifier (FK):

Entity:

Entity Identifier (FK):

Entity Definition Date (FK):

Entity Kind-of Definition Date (FK):

Entity Kind-of Identifier (FK):

Entity Identifier (FK):

Entity Definition Date (FK):

Entity-Attribute:

Entity Identifier (FK):

Entity Definition Date (FK):

Attribute Definition Date (FK):

Attribute Identifier (FK):

Attribute-Field-Correspondence:

Attribute-Entity Definition Date (FK):

Attribute Definition Date (FK):

PRELIMINARY DRAFT COPY

Field-Entity Definition Date (FK):

Information-Field Identifier (FK):

Attribute Identifier (FK):

Information Identifier (FK):

Entity Identifier (FK):

Information-Field:

Information-Field Identifier:

Information Definition Date (FK):

Information Identifier (FK):

Information-Field Data Type:

Information-Field Unit of Measure:

Information-Field Maximum Value:

Information-Field Minimum Value:

Information-Field Optional:

Information-Field Repeatable:

Information-Field-Correspondence:

First-Entity Definition Date (FK):

First-Field Identifier (FK):

Second-Entity Definition Date (FK):

Second-Field Identifier (FK):

Information Identifier (FK):

E-14. Interaction

Related Data Structures:

Interaction:

Interaction Identifier:

PRELIMINARY DRAFT COPY

Receiver-Task Identifier (FK):

Sender-Task Identifier (FK):

Interaction-Stereotype Identifier (FK):

Entity Identifier (FK):

Entity Definition Date (FK):

Interaction Quantity:

Interaction-Stereotype:

Interaction-Stereotype Identifier:

Interaction-Stereotype Name:

Interaction-Stereotype Definition:

{ Assignment (Initiate mission/task sequence for subordinate), Attack (Send something that will cause damage or casualties), Collision (Attempt to occupy same location), Delegation (Shift command of subordinate unit), Land (Cause to return to surface mode from flight), Launch (Transition an entity from surface mode to dynamic flight), Resupply (Provide additional supply of expendable materiel), Transfer (Shift control of materiel or facility), Transmission (Send information or disinformation)}

Task:

Task Identifier:

Configuration-Unit Identifier (FK):

Task Name:

Task Description:

Entity Subject Identifier (FK):

Entity Subject Definition Date (FK):

Action Identifier (FK):

Action Definition Date (FK):

Entity Direct-Object Identifier (FK):

Entity Direct-Object Definition Date (FK):

PRELIMINARY DRAFT COPY

Parent-Task Identifier (FK):

Entity:

Entity Identifier (FK):

Entity Definition Date (FK):

Entity Kind-of Definition Date (FK):

Entity Kind-of Identifier (FK):

Entity Identifier (FK):

Entity Definition Date (FK):

E-15. Metadata

Related Data Structures:

Configuration-Unit:

Configuration-Unit Identifier:

Configuration-Unit Name:

Configuration-Unit Version:

Configuration-Unit Description:

Configuration-Unit Access Authority:

Configuration-Unit Creation Date:

Configuration-Unit Registration Date:

Classification Identifier (FK):

Usage-Type Identifier (FK):

Usage-Record Identifier (FK):

Examination:

Examination Date:

Configuration-Unit Identifier (FK):

Examination-Type Identifier (FK):

PRELIMINARY DRAFT COPY

Examiner Identifier (FK):

Sponsor Identifier (FK):

Examination-Sponsor Identifier:

Classification:

Classification Identifier:

Classification Name:

Classification Description:

{Unclassified, Confidential, Secret, Top Secret, No Foreign, NATO}

Usage-Type:

Usage-Type Identifier:

Usage-Type Name:

Usage-Type Definition:

{Unlimited, No Expiry, Time Period, Per Access List, Per Access, ... }

Usage-Record:

Usage-Record Identifier:

Usage-Record Name:

Usage-Record Definition:

{Per Usage, Daily Summary by User, Monthly Summary by User, Monthly Hit Report, None}

Usage:

Usage Date Time Group:

Configuration-Unit Identifier (FK):

User Identifier (FK):

Usage Hits Quantity:

User:

User Identifier:

PRELIMINARY DRAFT COPY

User Name:

User Description:

User POC:

User Voice:

User FAX:

User Electronic:

User Pager:

User Cellular:

User USPS Line 1:

User USPS Line 2:

User USPS Line 3:

User USPS City:

User USPS State:

User USPS Zip:

User Other Line 1:

User Other Line 2:

User Other Line 3:

User Other City:

User Other State:

User Other Zip:

User-User-Type:

User Identifier (FK):

User-Type Identifier (FK):

Configuration-Unit Identifier (FK):

User-Type:

PRELIMINARY DRAFT COPY

User-Type Identifier:

User-Type Definition:

{Examiner, Extractor, Producer, Reader, Repository Manager, Sponsor}

E-16. Network

Related Data Structures:

Network:

Network Definition Date (FK):

Network Identifier (FK):

Network-Organization:

Network Definition Date (FK):

Network Identifier (FK):

Organization Definition Date (FK):

Organization Identifier (FK):

Organization:

Organization Definition Date (FK):

Organization Identifier (FK):

Network-Network:

Network Definition Date (FK):

Network Identifier (FK):

Network-Equipment:

Equipment Definition Date (FK):

Network Definition Date (FK):

Equipment Identifier (FK):

Network Identifier (FK):

Equipment:

PRELIMINARY DRAFT COPY

Equipment Definition Date (FK):

Equipment Identifier (FK):

E-17. Process Sequence

Related Data Structures:

Process-Sequence:

Process-Package Identifier (FK):

Parent-Process Identifier (FK):

First-Sub-Process Identifier (FK):

Second-Sub-Process Identifier (FK):

First-Part Identifier (FK):

Second-Part Identifier (FK):

Process-Sequence First Part Cardinality:

Process-Sequence Second Part Cardinality:

Process-Sequence Iteration Description:

Process:

Process-Package Identifier (FK):

Process Identifier:

Configuration-Unit Identifier (FK):

Process Name:

Process Description:

First-Part:

First-Part Identifier:

First-Part Name:

First-Part Definition:

{Start, End, Iteration, Concurrent}

PRELIMINARY DRAFT COPY

Second-Part:

Second-Part Identifier:

Second-Part Name:

Second-Part Description:

{ Prior, Start, During, End, After }

E-18. Reference

Related Data Structures:

Reference:

Reference Identifier:

Reference Name:

Reference Description:

Reference Location:

Reference ADS Identifier:

Examination-Reference:

Reference Identifier (FK):

Examination Date (FK):

Configuration-Unit Identifier (FK):

Examination:

Examination Date:

Configuration-Unit Identifier (FK):

Examination-Type Identifier (FK):

Examiner Identifier (FK):

Sponsor Identifier (FK):

Examination-Sponsor Identifier:

Use-Case-Package-Reference:

PRELIMINARY DRAFT COPY

Reference Identifier (FK):

Use-Case-Package Identifier (FK):

Use-Case-Package:

Use-Case-Package Identifier:

Configuration-Unit Identifier (FK):

Use-Case-Package Name:

Use-Case-Package Description:

Use-Case-Package Containing Identifier (FK):

Use-Case-Reference:

Reference Identifier (FK):

Use-Case Identifier (FK):

Use-Case-Package Identifier (FK):

Use-Case:

Use-Case Identifier:

Use-Case-Package Identifier (FK):

Entity Stimulating Identifier (FK):

Entity Stimulating Definition Date (FK):

Entity Responding Identifier (FK):

Entity Responding Definition Date (FK):

Use-Case Name:

Use-Case Purpose:

Use-Case Description:

Configuration-Unit Identifier (FK):

Use-Case-Instance-Reference:

Reference Identifier (FK):

PRELIMINARY DRAFT COPY

Use-Case Identifier (FK):

Use-Case-Package Identifier (FK):

Use-Case-Instance Identifier (FK):

Use-Case-Instance:

Use-Case-Package Identifier (FK):

Use-Case Identifier (FK):

Use-Case-Instance Identifier:

Configuration-Unit Identifier (FK):

Use-Case-Instance Name:

Use-Case-Instance Description:

Task-Reference:

Task Identifier (FK):

Reference Identifier (FK):

Task:

Task Identifier:

Configuration-Unit Identifier (FK):

Task Name:

Task Description:

Entity Subject Identifier (FK):

Entity Subject Definition Date (FK):

Action Identifier (FK):

Action Definition Date (FK):

Entity Direct-Object Identifier (FK):

Entity Direct-Object Definition Date (FK):

Parent-Task Identifier (FK):

PRELIMINARY DRAFT COPY

Data-Dictionary-Element-Reference:

Reference Identifier (FK):

Data-Dictionary-Element Identifier (FK):

Data-Dictionary-Element Definition Date (FK):

Data-Dictionary-Element:

Data-Dictionary-Element Identifier:

Data-Dictionary-Element Definition Date:

Data-Dictionary Identifier (FK):

Data-Dictionary-Element Name:

Data-Dictionary-Element Definition:

Data-Dictionary-Element Definition Authority:

Data-Dictionary-Element Type:

Data-Dictionary-Element-Sponsor Identifier (FK):

E-19. State Transition

Related Data Structures:

State-Transition:

State-Set Identifier (FK):

State-Transition Identifier:

Entity Identifier (FK):

Entity Definition Date (FK):

Attribute Definition Date (FK):

Attribute Identifier (FK):

Task Identifier (FK):

State-Transition Entry Value:

State-Transition Exit Value:

PRELIMINARY DRAFT COPY

State-Variable:

State-Set Identifier (FK):

Entity Identifier (FK):

Entity Definition Date (FK):

Attribute Definition Date (FK):

Attribute Identifier (FK):

Entity-Attribute:

Entity Identifier (FK):

Entity Definition Date (FK):

Attribute Definition Date (FK):

Attribute Identifier (FK):

State-Set:

State-Set Identifier:

State-Set Name:

State-Set Definition:

Event:

Task Identifier (FK):

State-Set Identifier (FK):

Event-Type Identifier (FK):

Event-Type:

Event-Type Identifier:

Event-Type Description:

{Task Begin, Task End, Continuous}

Task:

Task Identifier:

PRELIMINARY DRAFT COPY

Configuration-Unit Identifier (FK):

Task Name:

Task Description:

Entity Subject Identifier (FK):

Entity Subject Definition Date (FK):

Action Identifier (FK):

Action Definition Date (FK):

Entity Direct-Object Identifier (FK):

Entity Direct-Object Definition Date (FK):

Parent-Task Identifier (FK):

E-20. Task

Related Data Structures:

Task:

Task Identifier:

Configuration-Unit Identifier (FK):

Task Name:

Task Description:

Entity Subject Identifier (FK):

Entity Subject Definition Date (FK):

Action Identifier (FK):

Action Definition Date (FK):

Entity Direct-Object Identifier (FK):

Entity Direct-Object Definition Date (FK):

Parent-Task Identifier (FK):

Entity:

PRELIMINARY DRAFT COPY

Entity Identifier (FK):

Entity Definition Date (FK):

Entity Kind-of Definition Date (FK):

Entity Kind-of Identifier (FK):

Entity Identifier (FK):

Entity Definition Date (FK):

Action:

Action Identifier (FK):

Action Definition Date (FK):

Subject Identifier (FK):

Subject (FK):

Verb Identifier (FK):

Verb Definition Date (FK):

Direct-Object Identifier (FK):

Direct-Object Definition Date (FK):

Task-Phrase:

Task Identifier (FK):

Task-Phrase Sequence Identifier:

Entity Identifier (FK):

Entity Definition Date (FK):

Modifier Identifier (FK):

Modifier:

Modifier Identifier:

Modifier Name:

Modifier Definition:

PRELIMINARY DRAFT COPY

{ And, For, From, Of, Or, To, Using, With }

Condition-Clause:

Task Identifier (FK):

Use-Case Identifier (FK):

Use-Case-Package Identifier (FK):

Use-Case-Condition Identifier (FK):

Condition-Clause-Kind Identifier (FK):

Condition-Clause-Type Identifier (FK):

Condition-Clause Value:

Condition-Clause-Type:

Condition-Clause-Type Identifier:

Condition-Clause-Type Name:

Condition-Clause-Type Definition:

{ If, When, While }

Condition-Clause-Kind:

Condition-Clause-Kind Identifier:

Condition-Clause-Kind Name:

Condition-Clause-Kind Description:

{ Start, Terminate, Interrupt }

Use-Case-Condition:

Use-Case-Package Identifier (FK):

Use-Case Identifier (FK):

Use-Case-Condition Identifier:

Condition Identifier (FK):

Entity Identifier (FK):

PRELIMINARY DRAFT COPY

Entity Definition Date (FK):

Attribute Identifier (FK):

Attribute Definition Date (FK):

Relation Identifier (FK):

On-Terminate-Start:

Task Identifier (FK):

Use-Case Identifier (FK):

Use-Case-Package Identifier (FK):

Use-Case-Condition Identifier (FK):

Next-Task Identifier (FK):

E-21. Task Sequence

Related Data Structures:

Task Sequence:

Use-Case Identifier (FK):

First-Task Identifier (FK):

Second-Task Identifier (FK):

First-Part Identifier (FK):

Use-Case-Package Identifier (FK):

Second-Part Identifier (FK):

Task-Sequence First Task Cardinality:

Task-Sequence Second Task Cardinality:

Task-Sequence Iteration:

Use-Case:

Use-Case Identifier:

Use-Case-Package Identifier (FK):

PRELIMINARY DRAFT COPY

Entity Stimulating Identifier (FK):

Entity Stimulating Definition Date (FK):

Entity Responding Identifier (FK):

Entity Responding Definition Date (FK):

Use-Case Name:

Use-Case Purpose:

Use-Case Description:

Configuration-Unit Identifier (FK):

Task:

Task Identifier:

Configuration-Unit Identifier (FK):

Task Name:

Task Description:

Entity Subject Identifier (FK):

Entity Subject Definition Date (FK):

Action Identifier (FK):

Action Definition Date (FK):

Entity Direct-Object Identifier (FK):

Entity Direct-Object Definition Date (FK):

Parent-Task Identifier (FK):

First-Part:

First-Part Identifier:

First-Part Name:

First-Part Definition:

{Start, End, Iteration, Concurrent}

PRELIMINARY DRAFT COPY

Second-Part:

Second-Part Identifier:

Second-Part Name:

Second-Part Description:

{ Prior, Start, During, End, After }

E-22. Use Case

Related Data Structures:

Use-Case:

Use-Case Identifier:

Use-Case-Package Identifier (FK):

Entity Stimulating Identifier (FK):

Entity Stimulating Definition Date (FK):

Entity Responding Identifier (FK):

Entity Responding Definition Date (FK):

Use-Case Name:

Use-Case Purpose:

Use-Case Description:

Configuration-Unit Identifier (FK):

Use-Case-Instance:

Use-Case-Package Identifier (FK):

Use-Case Identifier (FK):

Use-Case-Instance Identifier:

Configuration-Unit Identifier (FK):

Use-Case-Instance Name:

Use-Case-Instance Description:

PRELIMINARY DRAFT COPY

Use-Case-Measure-of-Performance:

Use-Case-Package Identifier (FK):

Use-Case Identifier (FK):

Use-Case-Measure-of-Performance Identifier:

Measure-Of-Performance Identifier (FK):

Second-Attribute Identifier (FK):

First-Attribute Identifier (FK):

Second-Attribute Definition Date (FK):

First-Attribute Definition Date (FK):

Operator Identifier (FK):

Unit-Of-Measure Identifier (FK):

Unit-of-Measure:

Unit-of-Measure Identifier:

Unit-of-Measure Name:

Unit-of-Measure Description:

{Hours, Minutes, Seconds, Miles, Feet, Kilometers, Meters, Feet per Second,
Meters per Second, Percent Successful Identification, Percent Successful Attack,
...}

Operator:

Operator Identifier:

Operator Name:

Operator Definition:

{Add, Subtract, Multiply, Divide, Percent, Modulo, Negate}

Measure-of-Performance:

Measure-of-Performance Identifier:

Measure-of-Performance Name:

PRELIMINARY DRAFT COPY

Measure-of-Performance Description:

Measure-of-Performance UJTL Identifier:

Entity:

Entity Identifier (FK):

Entity Definition Date (FK):

Entity Kind-of Definition Date (FK):

Entity Kind-of Identifier (FK):

Entity Identifier (FK):

Entity Definition Date (FK):

Use-Case-Condition:

Use-Case-Package Identifier (FK):

Use-Case Identifier (FK):

Use-Case-Condition Identifier:

Condition Identifier (FK):

Entity Identifier (FK):

Entity Definition Date (FK):

Attribute Identifier (FK):

Attribute Definition Date (FK):

Relation Identifier (FK):

Condition:

Condition Identifier:

Condition Name:

Condition Description:

Condition UJTL Identifier:

Entity-Attribute:

PRELIMINARY DRAFT COPY

Entity Identifier (FK):

Entity Definition Date (FK):

Attribute Definition Date (FK):

Attribute Identifier (FK):

Attribute:

Attribute Identifier (FK):

Attribute Definition Date (FK):

Attribute Kind-of Definition Date (FK):

Attribute Identifier (FK):

Attribute Definition Date (FK):

E-23. Use Case Instance

Related Data Structures:

Use-Case-Instance:

Use-Case-Package Identifier (FK):

Use-Case Identifier (FK):

Use-Case-Instance Identifier:

Configuration-Unit Identifier (FK):

Use-Case-Instance Name:

Use-Case-Instance Description:

Standard:

Use-Case-Package Identifier (FK):

Use-Case Identifier (FK):

Use-Case-Instance Identifier (FK):

Use-Case-Measure-of-Performance Identifier (FK):

Standard Value:

PRELIMINARY DRAFT COPY

Use-Case-Measure-of-Performance:

- Use-Case-Package Identifier (FK):
- Use-Case Identifier (FK):
- Use-Case-Measure-of-Performance Identifier:
- Measure-Of-Performance Identifier (FK):
- Second-Attribute Identifier (FK):
- First-Attribute Identifier (FK):
- Second-Attribute Definition Date (FK):
- First-Attribute Definition Date (FK):
- Operator Identifier (FK):
- Unit-Of-Measure Identifier (FK):

Condition-Status:

- Use-Case-Package Identifier (FK):
- Use-Case Identifier (FK):
- Use-Case-Instance Identifier (FK):
- Use-Case-Condition Identifier (FK):
- Condition-Status Value:

Use-Case-Condition:

- Use-Case-Package Identifier (FK):
- Use-Case Identifier (FK):
- Use-Case-Condition Identifier:
- Condition Identifier (FK):
- Entity Identifier (FK):
- Entity Definition Date (FK):
- Attribute Identifier (FK):

PRELIMINARY DRAFT COPY

Attribute Definition Date (FK):

Relation Identifier (FK):

E-24. Use Case Package

Related Data Structures:

Use-Case-Package:

Use-Case-Package Identifier:

Configuration-Unit Identifier (FK):

Use-Case-Package Name:

Use-Case-Package Description:

Use-Case-Package Containing Identifier (FK):

Configuration-Unit:

Configuration-Unit Identifier:

Configuration-Unit Name:

Configuration-Unit Version:

Configuration-Unit Description:

Configuration-Unit Access Authority:

Configuration-Unit Creation Date:

Configuration-Unit Registration Date:

Classification Identifier (FK):

Usage-Type Identifier (FK):

Usage-Record Identifier (FK):

Use-Case:

Use-Case Identifier:

Use-Case-Package Identifier (FK):

Entity Stimulating Identifier (FK):

PRELIMINARY DRAFT COPY

Entity Stimulating Definition Date (FK):

Entity Responding Identifier (FK):

Entity Responding Definition Date (FK):

Use-Case Name:

Use-Case Purpose:

Use-Case Description:

Configuration-Unit Identifier (FK):

Uses:

Task Identifier (FK):

Use-Case Identifier (FK):

Use-Case-Package Identifier (FK):

Task:

Task Identifier:

Configuration-Unit Identifier (FK):

Task Name:

Task Description:

Entity Subject Identifier (FK):

Entity Subject Definition Date (FK):

Action Identifier (FK):

Action Definition Date (FK):

Entity Direct-Object Identifier (FK):

Entity Direct-Object Definition Date (FK):

Parent-Task Identifier (FK):

E-25. User

Related Data Structures:

PRELIMINARY DRAFT COPY

User:

User Identifier:

User Name:

User Description:

User POC:

User Voice:

User Fax:

User Electronic:

User Pager:

User Cellular:

User USPS Line 1:

User USPS Line 2:

User USPS Line 3:

User USPS City:

User USPS State:

User USPS Zip:

User Other Line 1:

User Other Line 2:

User Other Line 3:

User Other City:

User Other State:

User Other Zip:

User-User-Type:

User Identifier (FK):

User-Type Identifier (FK):

PRELIMINARY DRAFT COPY

Configuration-Unit Identifier (FK):

User-Type:

User-Type Identifier:

User-Type Definition:

{Examiner, Extractor, Producer, Reader, Repository Manager, Sponsor}

Configuration-Unit:

Configuration-Unit Identifier:

Configuration-Unit Name:

Configuration-Unit Version:

Configuration-Unit Description:

Configuration-Unit Access Authority:

Configuration-Unit Creation Date:

Configuration-Unit Registration Date:

Classification Identifier (FK):

Usage-Type Identifier (FK):

Usage-Record Identifier (FK):

PRELIMINARY DRAFT COPY

A

accreditation..... 11, 14, 63, 68, 75, 103
action13, 14, 16, 17, 18, 26, 27, 59, 61, 68, 70, 72,
80, 83
activity10, 15, 16, 17, 20, 23, 25-27, 28, 29, 30, 31,
32, 34, 35, 36, 39, 41, 42, 53, 55, 57, 58, 59, 64,
68, 69, 71, 72, 73, 78, 82-84, 86, 90, 91, 92, 97,
108, 115, 116, 119, 120, 129
actor 17, 19, 23, 29, 37, 38-39, 52, 68
adjacency..... 46-47, 50, 53, 54, 61, 69, 85, 99
agent18, 19, 25, 26, 28, 31, 39, 40, 43, 46, 47, 48, 49,
52, 53, 54, 65, 69, 70, 71, 72, 81, 96
aggregation..... 20, 56, 69
assignment..... 23, 52-53, 55, 61, 69, 70, 99, 107
association13, 15, 20, 46-50, 52, 53, 54, 56, 61, 69,
71, 84-86, 99
attack..18, 46, 49, 52, 53, 55, 61, 69, 70, 99, 107, 123
attribute20, 23, 24, 30, 31, 36, 38, 39, 41, 44, 53, 56,
58, 59, 61, 65-66, 68, 69, 70, 72, 87, 90, 91, 94,
97, 106, 117, 118, 122, 123, 124, 125, 126, 127

B

benchmark..... 12, 18, 61, 63, 103
Bezant Object Technology 73
Booch, Grady 73, 75

C

capabiliy.....34, 41-42, 45, 46, 69, 78, 79, 86
CASE 52, 73, 74
certification11, 12, 61, 62, 63, 69, 75, 98, 102, 103-
104
civil environment 8, 24, 35, 36, 76, 80-81
class.....20, 21, 23, 39, 40, 45, 66
classification25, 56, 57, 60, 61, 63, 66, 79, 89, 99,
101, 105, 109, 128
CMMS8, 9, 10, 11, 12, 13, 14, 15, 16, 18, 20, 21, 22,
23, 24, 25, 27, 30, 31, 32, 35, 36, 38, 39, 41, 42,
43, 44, 45, 46, 50, 52, 55, 56, 57, 58, 59, 60, 61,
62, 63, 64, 65, 66, 68, 69, 70, 73, 74, 75, 82
collaboration46, 47, 50, 61, 69, 85, 99
Collaboration Diagrams 32
collision.....52, 53, 55, 61, 69, 70, 99, 107
command46, 47, 48, 50, 52, 53, 55, 56, 61, 69, 70,
71, 75, 78, 79, 80, 85, 99, 107
common syntax and semantics..... 13, 14, 27
communication44, 46, 47, 50, 54, 61, 66, 69, 71, 78,
79, 85, 99
conceptual model .8, 10, 11, 12, 13, 14, 15, 21, 56, 65
conceptual object 38, 45, 64, 65, 69, 81
conceptual object model (COM)11, 56, 64, 65, 69, 74

condition16, 19, 23, 24, 29, 31, 35-38, 52, 57, 61, 64,
67, 69, 72, 73, 76, 77, 78, 79, 80, 81, 87-88, 100,
121-122, 124-125, 126, 127
conditional.....26, 29, 32, 34, 36, 37, 60
configuration unit8, 11, 23, 24, 25, 27, 29, 31, 35, 36,
37, 38, 50, 55, 56, 57, 58, 60, 63, 64, 65, 66, 68,
70, 83, 85, 88-90, 101, 103, 104-105, 107, 108-
109, 110, 113, 114, 116, 117, 119, 123, 125, 127-
128
control46, 47, 50, 53, 54, 55, 61, 69, 70, 71, 72, 78,
79, 85, 99, 107
courses of action 16, 17, 18

D

data dictionary13, 14, 25, 29, 41, 43, 55, 57-59, 61,
64, 70, 90-93, 94, 95, 97, 115
delegation.....52, 53, 55, 61, 70, 99, 107
Department of Defense (DoD) 23, 36, 37, 74, 75
DMSO 8, 74, 75
domain object model (DOM)..... 37, 74

E

EATI 13, 14
entity8, 13, 14, 17, 18, 23, 24, 25, 26, 27, 28, 29, 30,
31, 32, 33, 35, 36, 37, 38-45, 46, 47, 48, 49, 50,
52, 53, 54, 55, 57, 58, 59, 61, 63, 64, 65, 66, 67,
68, 69, 70, 71, 72, 73, 77, 81-82, 83-84, 85, 86,
87, 89, 90, 91, 92, 93-95, 97, 99, 100, 105, 106,
107, 108, 113, 114, 115, 116, 117, 118, 119, 120,
122, 123, 124, 125, 126, 127, 128, 129
entity type.....39, 42-43, 44, 70, 95-97
enumeration.46, 57, 58, 59, 60-61, 70, 90, 91, 97-101
environment8, 9, 10, 15, 18, 24, 35, 36, 37, 40, 69,
72, 73, 76-81
equipment23, 36, 40, 41, 42, 43, 45, 46, 61, 69, 70,
71, 95, 96, 101, 112
event11, 17, 18, 20, 26, 43, 50, 65, 68, 70, 82, 118
examination12, 56, 60, 61-63, 64, 70, 98, 101-104,
109, 112-113

F

facility40, 42, 43, 49, 54, 55, 61, 70, 72, 80, 82, 96,
99, 107
feature.....40, 42, 43, 70, 73, 82, 96
fidelity.....9, 13, 40, 66, 70, 104-105

G

generic task 29, 30, 70
Graham, Ian..... 74, 75

H

Harel, David..... 73

PRELIMINARY DRAFT COPY

I

I-logix..... 73
 implementation object model (IOM) 11, 74
 in range 46-47, 53
 inactive entity..... 39, 40, 70
 inheritance..... 21, 22, 40, 46, 56, 70, 73
 information 9, 10, 11, 12, 13, 15, 16, 17, 23, 24, 28,
 29, 31, 34, 37, 40, 42, 43-44, 47, 48, 52, 54, 55,
 56, 59, 61, 63, 65, 66, 69, 70, 72, 73, 78, 79, 81,
 95-96, 99, 105-107
 interaction 13, 14, 15, 19, 20, 31, 32, 34, 46, 47, 50-
 55, 61, 67, 69, 70, 71, 72, 99, 107-108

J

Jacobson, Ivar 73
 Joint Application Development (JAD)..... 10, 74
 JWARS 37, 74, 75

K

knowledge acquisition..... 9, 10, 14, 18, 19, 45
 knowledge element (KL) .8, 14, 16, 22, 37, 56, 58, 74
 knowledge engineering 10, 18, 45

L

land..... 52, 53, 55, 61, 70, 71, 76, 99, 107
 launch..... 52, 53-54, 55, 61, 64, 70, 71, 99, 107
 lesser regional contingency (LRC)..... 37, 74
 line of sight 46
 line of support 46

M

materiel 16, 40, 42, 43, 47, 49, 53, 54, 55, 61, 69, 70,
 71, 72, 79, 80, 82, 96, 99, 107
 measures of performance (MOP) 23, 36-38, 71, 73,
 123, 124, 126
 metadata 59-60, 71, 108-111
 military environment..... 8, 24, 35, 36, 76, 78-80
 military operations mission space (MOMS) 10, 11,
 12, 13, 15, 24, 25, 27, 38, 46, 55, 73, 74
 mission 9, 17, 18, 23, 38, 54, 55, 61, 71, 74, 78, 99,
 107
 mission space 15, 16
 model 8, 9, 11, 13, 14, 20, 22, 23, 61, 62, 63, 64, 65,
 66, 68, 69, 70, 71, 72, 73, 74, 75
 major regional contingency (MRC) 37, 74
 multiple inheritance..... 21

N

network 9, 40, 42, 43, 44-45, 70, 71, 96, 111-112
 nominal tasks 31

O

Object Modeling Technique (OMT) 73, 75
 object-oriented 18, 20, 22, 39, 45, 73, 75
 Objectory 73

objects 11, 12, 17, 20, 21, 23, 30, 39, 45, 47, 53, 55,
 56, 61, 64, 65, 67, 69, 70, 72, 73, 74, 75, 78, 81
 occupancy 46, 47-48, 50, 61, 69, 71, 85, 99
 occupation 46, 48, 50, 61, 69, 71, 85, 99
 operation 16, 18, 20, 21, 23, 28, 39, 46, 48, 50, 52, 53,
 54, 61, 67, 69, 71, 72, 80, 85, 97-98, 99
 opposition..... 18, 46, 47, 48, 50, 61, 69, 71, 85, 99
 organization 25, 36, 40, 41, 42, 43, 45, 46, 48, 50, 54,
 61, 65, 69, 70, 71, 73, 79, 80, 81, 82, 85, 95, 96,
 97, 99, 101, 111, 112

P

phrase..... 25, 26, 27, 28, 31, 68, 84, 120
 physical environment 8, 18, 24, 35, 36, 76-78
 player 17, 18, 19, 20, 39, 40, 71
 polymorphism 21
 possession 46, 48, 49, 50, 61, 69, 71, 85, 99
 producer 8, 12, 25, 57, 60, 61, 63, 71, 101, 105, 109,
 111, 128
 process 19, 20, 23, 26, 27, 32, 39, 62, 71, 72, 83

R

rapid application development (RAD) 10, 75
 rank 45, 82
 Rational 73, 75
 reference..... 23, 29, 63-64, 71, 105, 112-115
 resolution..... 9, 13, 37, 66
 resupply..... 52, 54, 55, 61, 70, 71, 80, 99, 107
 reusability..... 9, 11, 13, 14, 15, 29
 role 16, 17, 18, 19, 39, 40, 45, 71, 72, 81
 Rumbaugh, James 20, 73, 75

S

scenario 8, 10, 16, 17
 sequence 10, 23, 25, 27, 29, 31, 32-35, 52, 61, 72, 84,
 99, 100, 107, 115-117, 120
 Sheehan, Jack..... 75
 subject matter expert (SME) 10, 15, 18, 19, 75
 SOMA 73, 75
 SOMATiK 73, 75
 specific task..... 30, 72
 specification 12, 14, 18, 19, 25, 27, 28, 30, 31, 32, 34,
 37, 39, 50, 52, 61, 62, 63, 72, 73, 103
 standard 13, 14, 19, 24, 36-38, 47, 61, 69, 72, 73, 126
 state 10, 18, 19, 20, 37, 38, 40, 50, 55, 67-68, 70, 72,
 73, 117, 118
 StateMate 73
 state transition 19, 67-68, 72, 117-119
 state variables..... 39, 58, 67, 68, 72, 118
 subclass 56, 70
 substantive tasks..... 31
 subtask 30, 72
 superclass 56, 70
 supply..... 46, 49, 50, 54, 55, 61, 69, 72, 85, 99, 107
 support 46, 47, 49, 50, 61, 69, 72, 85, 99
 SVD[PI]* 13, 14, 25, 26, 28, 29

PRELIMINARY DRAFT COPY

system9, 10, 11, 12, 16, 20, 23, 39, 44, 56, 61, 63, 64,
65, 66, 70, 71, 72, 73, 74-76, 78, 103, 105

T

task10, 13, 14, 15, 16, 17, 18, 21, 23, 25, 26, 27-31,
32, 33, 34, 35, 36, 37, 38, 39, 40, 46, 47, 50, 51,
52, 53, 55, 57, 58, 61, 64, 65, 67, 68, 69, 70, 71,
72, 73, 74, 75, 76, 77, 78, 80, 83, 90, 99, 100, 107-
108, 114-115, 116, 117, 118, 119-122, 128-129
task step..... 30, 72
traceability..... 22, 23, 63
transfer32, 33, 52, 54, 55, 56, 61, 70, 72, 99, 107
transmission47, 52, 54, 55, 61, 70, 72, 99, 107

U

UJTL24, 29, 75, 76, 77, 78, 80, 81, 124, 125
Unified Modeling Language (UML).....32, 52, 73, 75
use case17, 18, 19, 20, 23-25, 28, 29, 30, 31, 32, 35,
36, 37, 38, 39, 52, 53, 57, 64, 69, 71, 72, 73, 87-
88, 89, 113-114, 115, 116-117, 121-125
use case instance24, 36-38, 39, 40, 57, 64, 73, 88, 89-
90, 114, 123, 125-127
use case package20, 24, 25, 31, 35, 36, 38, 39, 50, 56,
57, 64, 70, 73, 85, 87, 88, 89, 113, 114, 116, 117,
121, 122, 123, 124, 125, 126, 127-129

V

validation11, 12, 14, 59, 61, 62, 63, 73, 75, 98, 102,
103, 104
verb25, 26, 27, 28, 29, 31, 42, 55, 57, 58, 59, 61, 68,
73, 82, 84, 86, 90, 91, 92, 97, 120
verification11, 12, 14, 59, 61, 62, 63, 73, 75, 98, 102,
103, 104
virtual classes 21, 39, 45

W

WARSIM 75
weather 37, 40, 76, 77-78, 82

PRELIMINARY DRAFT COPY